Polyhedral sets, lattice points, optimizing compilers and computer algebra

Marc Moreno Maza[1]

[1]Ontario Research Center for Computer Algebra, UWO, London, Ontario

March 14, 2025(version 2)

Plan

Acknowledgements

- Many thanks to the JNCF 2025 organizers, who gave me the opportunity to be back in Luminy 20 years after.

Acknowledgements

- Many thanks to the JNCF 2025 organizers, who gave me the opportunity to be back in Luminy 20 years after.
- This tutorial is based on research projects in which many of my former and current PhD students have played an essential role. By alphabetic order: Xiaohui Chen, Rui-Juan Jing Yuzhuo Lei, Christopher Maligec, Chirantan Mukherjee, Delaram Talaashrafi, Linxiao Wang and Ning Xie.

Acknowledgements

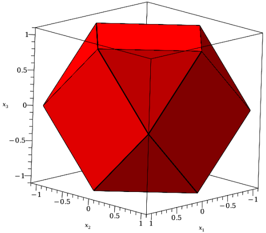
- Many thanks to the JNCF 2025 organizers, who gave me the opportunity to be back in Luminy 20 years after.
- This tutorial is based on research projects in which many of my former and current PhD students have played an essential role. By alphabetic order: Xiaohui Chen, Rui-Juan Jing Yuzhuo Lei, Christopher Maligec, Chirantan Mukherjee, Delaram Talaashrafi, Linxiao Wang and Ning Xie.
- This tutorial is based on collaborations with Maplesoft, MIT/CSAIL, NVIDIA, Intel and IBM Canada, with funding support from Maplesoft, MITACS, IBM and NSERC of Canada.

Acknowledgements

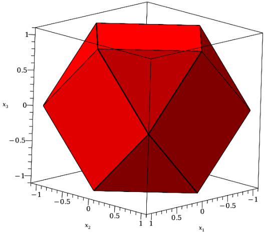
- Many thanks to the JNCF 2025 organizers, who gave me the opportunity to be back in Luminy 20 years after.
- This tutorial is based on research projects in which many of my former and current PhD students have played an essential role. By alphabetic order: Xiaohui Chen, Rui-Juan Jing Yuzhuo Lei, Christopher Maligec, Chirantan Mukherjee, Delaram Talaashrafi, Linxiao Wang and Ning Xie.
- This tutorial is based on collaborations with Maplesoft, MIT/CSAIL, NVIDIA, Intel and IBM Canada, with funding support from Maplesoft, MITACS, IBM and NSERC of Canada.
- Most of the algorithms presented in this tutorial are implemented in MAPLE's `PolyhedralSets` library.

$$
\text{null}\begin{cases}
-x_3 \le 1 \\
-x_1 - x_2 - x_3 \le 2 \\
-x_1 + x_2 - x_3 \le 2 \\
x_1 - x_2 - x_3 \le 2 \\
x_1 + x_2 - x_3 \le 2 \\
x_3 0 \le 1 \\
-x_1 - x_2 + x_3 \le 2 \\
-x_1 + x_2 + x_3 \le 2 \\
x_1 - x_2 + x_3 \le 2 \\
x_1 + x_2 + x_3 \le 2 \\
-x_2 0 \le 1 \\
x_2 \le 1 \\
-x_1 \le 1 \\
x_1 0 \le 1
\end{cases}
$$

$$
\text{null}\begin{cases}
-x_3 \le 1 \\
-x_1 - x_2 - x_3 \le 2 \\
-x_1 + x_2 - x_3 \le 2 \\
x_1 - x_2 - x_3 \le 2 \\
x_1 + x_2 - x_3 \le 2 \\
x_3 0 \le 1 \\
-x_1 - x_2 + x_3 \le 2 \\
-x_1 + x_2 + x_3 \le 2 \\
x_1 - x_2 + x_3 \le 2 \\
x_1 + x_2 + x_3 \le 2 \\
-x_2 0 \le 1 \\
x_2 \le 1 \\
-x_1 \le 1 \\
x_1 0 \le 1
\end{cases}
$$

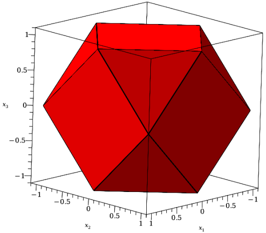


$$\text{null}\begin{cases} -x_3 \leq 1 \\ -x_1 - x_2 - x_3 \leq 2 \\ -x_1 + x_2 - x_3 \leq 2 \\ x_1 - x_2 - x_3 \leq 2 \\ x_1 + x_2 - x_3 \leq 2 \\ x_3 0 \leq 1 \\ -x_1 - x_2 + x_3 \leq 2 \\ -x_1 + x_2 + x_3 \leq 2 \\ x_1 - x_2 + x_3 \leq 2 \\ x_1 + x_2 + x_3 \leq 2 \\ -x_2 0 \leq 1 \\ x_2 \leq 1 \\ -x_1 \leq 1 \\ x_1 0 \leq 1 \end{cases}$$

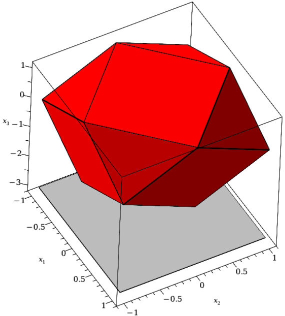


$$\text{null}\begin{cases} 0 \leq 1 + x_2 \\ 0 \leq 1 - x_2 \\ 0 \leq x_1 + 1 \\ 0 \leq 1 - x_1 \end{cases}$$

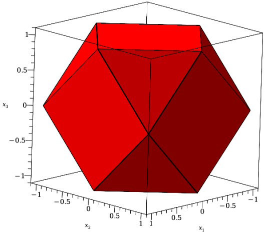
$$
\text{null}\left\{
\begin{array}{r}
-x_3 \leq 1 \\
-x_1 - x_2 - x_3 \leq 2 \\
-x_1 + x_2 - x_3 \leq 2 \\
x_1 - x_2 - x_3 \leq 2 \\
x_1 + x_2 - x_3 \leq 2 \\
x_3 0 \leq 1 \\
-x_1 - x_2 + x_3 \leq 2 \\
-x_1 + x_2 + x_3 \leq 2 \\
x_1 - x_2 + x_3 \leq 2 \\
x_1 + x_2 + x_3 \leq 2 \\
-x_2 0 \leq 1 \\
x_2 \leq 1 \\
-x_1 \leq 1 \\
x_1 0 \leq 1
\end{array}
\right.
$$



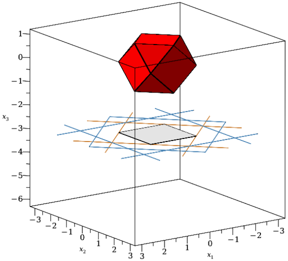
$$
\text{null}\left\{
\begin{array}{l}
0 \leq 1 + x_2 \\
0 \leq 1 - x_2 \\
0 \leq x_1 + 1 \\
0 \leq 1 - x_1
\end{array}
\right.
$$



$$\left\{\text{null}\begin{aligned}
-x_3 &\le 1\\
-x_1 - x_2 - x_3 &\le 2\\
-x_1 + x_2 - x_3 &\le 2\\
x_1 - x_2 - x_3 &\le 2\\
x_1 + x_2 - x_3 &\le 2\\
x_3 0 &\le 1\\
-x_1 - x_2 + x_3 &\le 2\\
-x_1 + x_2 + x_3 &\le 2\\
x_1 - x_2 + x_3 &\le 2\\
x_1 + x_2 + x_3 &\le 2\\
-x_2 0 &\le 1\\
x_2 &\le 1\\
-x_1 &\le 1\\
x_1 0 &\le 1
\end{aligned}\right.$$



$$\left\{\text{null}\begin{aligned}
0 &\le 1 + x_2\\
0 &\le 1 - x_2\\
0 &\le x_1 + 1\\
0 &\le 1 - x_1
\end{aligned}\right.$$



Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Application of FME: code generation

```
for(i=0; i<=n; i++){
   c[i] = 0; c[i+n] = 0;
   for(j=0; j<=n; j++)
      c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t, p) := (n - j, i + j)$.

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t,p) := (n-j, i+j)$.

$$\left\{ \text{null} \begin{array}{rr} 0 \leq & i \\ i \leq & n \\ 0 \leq & j \\ j \leq & n \\ t = & n-j \\ p = & i+j \end{array} \right.$$

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t, p) := (n - j, i + j)$.

$$\text{null}\begin{cases} 0 \le & i \\ i \le & n \\ 0 \le & j \\ j \le & n \\ t = & n - j \\ p = & i + j \end{cases}$$

FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t, p) := (n - j, i + j)$.

$$\left\{ \text{null} \begin{aligned} 0 &\le i \\ i &\le n \\ 0 &\le j \\ j &\le n \\ t &= n - j \\ p &= i + j \end{aligned} \right.$$

$$\left\{ \begin{aligned} i &= p + t - n \\ j &= -t + n \\ t &\ge \max(0, -p + n) \\ \text{null}\, t &\le \min(n, -p + 2n) \\ 0 &\le p \\ p &\le 2n \\ 0 &\le n. \end{aligned} \right.$$

FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields: $(t, p) := (n - j, i + j)$.

$$\left\{ \text{null} \begin{array}{rr} 0 \le & i \\ i \le & n \\ 0 \le & j \\ j \le & n \\ t = & n - j \\ p = & i + j \end{array} \right.$$

The new representation allows us to generate the multithreaded code.

$$\left\{ \text{null} \begin{array}{rr} i = & p + t - n \\ j = & -t + n \\ t \ge & \max(0, -p + n) \\ t \le & \min(n, -p + 2n) \\ 0 \le & p \\ p \le & 2n \\ 0 \le & n. \end{array} \right.$$

FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

```
parallel_for (p=0; p<=2*n; p++){
  c[p] = 0;
  for (t=max(0,n-p);
       t<=min(n,2*n-p);t++)
    c[p] += A[t+p-n] * B[n-t];
}
```

Dependence analysis yields: $(t, p) := (n - j, i + j)$.

The new representation allows us to generate the multithreaded code.

$$\text{null}\begin{cases} 0 \leq & i \\ i \leq & n \\ 0 \leq & j \\ j \leq & n \\ t = & n - j \\ p = & i + j \end{cases}$$

$$\text{null}\begin{cases} i = & p + t - n \\ j = & -t + n \\ t \geq & \max(0, -p + n) \\ t \leq & \min(n, -p + 2n) \\ 0 \leq & p \\ p \leq & 2n \\ 0 \leq & n. \end{cases}$$

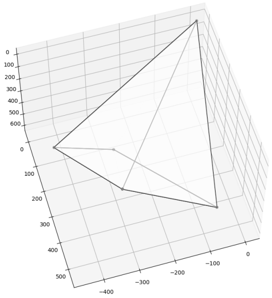
FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Application of FME: computing integer hulls (1/3)

The input polyhedral set:

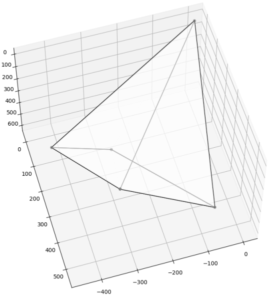
$$\begin{cases} -98877x_1 - 189663x_2 - 1798x_3 & \leq & 705915 \\ -10109x_1 - 5958x_2 - 14601x_3 & \leq & 31333 \\ -5405x_1 + 4965x_2 + 3870x_3 & \leq & 4303504 \\ 729x_1 - 117x_2 + 350x_3 & \leq & 4561 \\ 677x_1 + 465x_2 - 540x_3 & \leq & 3489 \end{cases}$$



Application of FME: computing integer hulls (1/3)

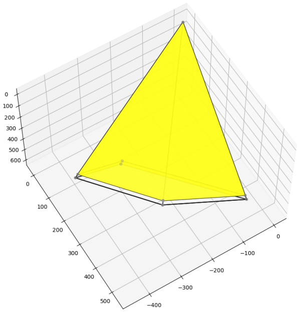
The input polyhedral set:

$$\left\{\begin{array}{rcl} -98877x_1 - 189663x_2 - 1798x_3 & \leq & 705915 \\ -10109x_1 - 5958x_2 - 14601x_3 & \leq & 31333 \\ -5405x_1 + 4965x_2 + 3870x_3 & \leq & 4303504 \\ 729x_1 - 117x_2 + 350x_3 & \leq & 4561 \\ 677x_1 + 465x_2 - 540x_3 & \leq & 3489 \end{array}\right.$$

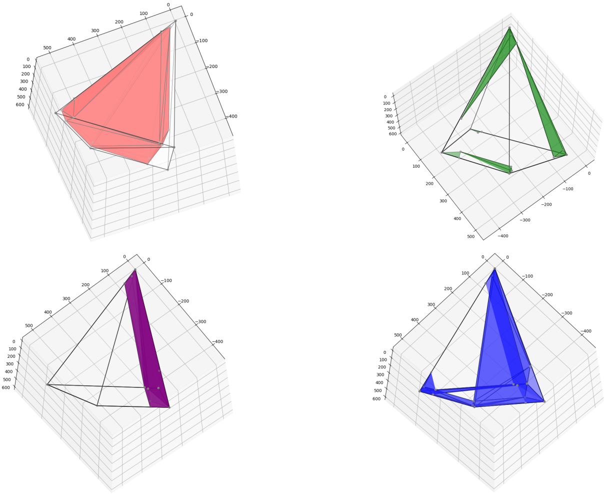


Normalization (leaves the integer hull unchanged):

$$\left\{\begin{array}{rcl} -98877x_1 - 189663x_2 - 1798x_3 & \leq & 705915 \\ -10109x_1 - 5958x_2 - 14601x_3 & \leq & 31333 \\ -1081x_1 + 993x_2 + 774x_3 & \leq & 860700 \\ 729x_1 - 117x_2 + 350x_3 & \leq & 4561 \\ 677x_1 + 465x_2 - 540x_3 & \leq & 3489 \end{array}\right.$$



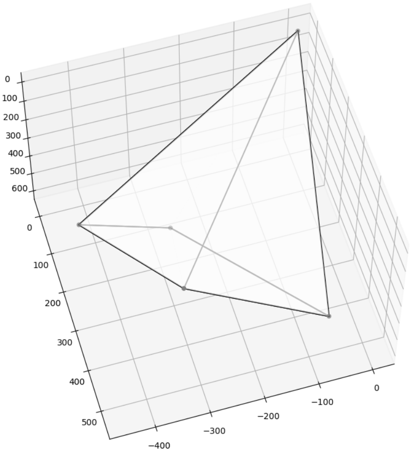
Application of FME: computing integer hulls (2/3)



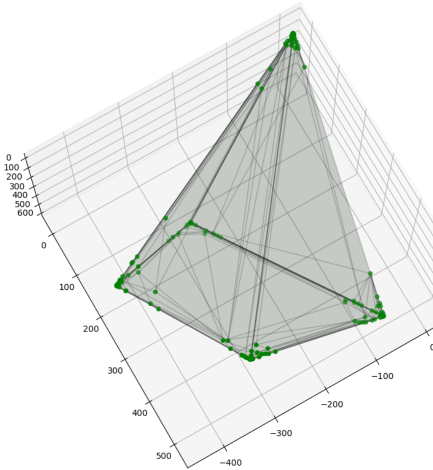
1. The **red** is an approximation of the integer hull of the input.
2. The integer hulls of border regions (green, blue, purple) are brute-force computed via FME.
3. Then `QuickHull` is applied to obtain the integer hull of the input.

Application of FME: computing integer hulls (3/3)

The input has only 5 vertices.



Its integer hull has 139 vertices.



All details are in https://ir.lib.uwo.ca/etd/8985/ and in https://doi.org/10.1007/978-3-031-14788-3_14

Application to dependence analysis

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] = A[(n * j - n + j - i - 1];
```

Application to dependence analysis

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] = A[(n * j - n + j - i - 1];
```

1. Can we parallelize the two `for-loops`?

Application to dependence analysis

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] = A[(n * j - n + j - i - 1];
```

1. Can we parallelize the two `for-loops`?
2. Is there data dependence between two different iterations of the nest?

Application to dependence analysis

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] = A[(n * j - n + j - i - 1];
```

1. Can we parallelize the two `for-loops`?
2. Is there data dependence between two different iterations of the nest?
3. Are there integer solutions to the following system of linear inequalities?

$$
\begin{cases}
\text{null} & \begin{aligned} 0 \leq & i_1 < n \\ i_1 + 1 \leq & j_1 < n \\ 0 \leq & i_2 < n \\ i_2 + 1 \leq & j_2 < n \\ i_1 \times n + j_1 = & n \times j_2 - n + j_2 - i_2 - 1 \end{aligned}
\end{cases}
$$

Delinearize the array accesses

Linearized one-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] =
        A[(n * j - n + j - i - 1];
```

Delinearize the array accesses

Linearized one-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] =
        A[(n * j - n + j - i - 1];
```

Delinearized multi-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    B[i][j] = B[j - 1][j - i - 1];
```

Delinearize the array accesses

$$\left\{ \text{null} \quad \begin{aligned} 0 \le & i_1 < n \\ i_1 + 1 \le & j_1 < n \\ 0 \le & i_2 < n \\ i_2 + 1 \le & j_2 < n \\ i_1 \times n + j_1 = & n \times j_2 - n + j_2 - i_2 - 1 \end{aligned} \right.$$

Linearized one-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] =
        A[(n * j - n + j - i - 1];
```

Delinearized multi-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    B[i][j] = B[j - 1][j - i - 1];
```

Delinearize the array accesses

Linearized one-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] =
        A[(n * j - n + j - i - 1];
```

$$\left\{ \text{null} \quad \begin{aligned} 0 &\le i_1 < n \\ i_1 + 1 &\le j_1 < n \\ 0 &\le i_2 < n \\ i_2 + 1 &\le j_2 < n \\ i_1 \times n + j_1 &= n \times j_2 - n + j_2 - i_2 - 1 \end{aligned} \right.$$

Delinearized multi-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    B[i][j] = B[j - 1][j - i - 1];
```

$$\left\{ \text{null} \quad \begin{aligned} 0 &\le i_1 < n \\ i_1 + 1 &\le j_1 < n \\ 0 &\le i_2 < n \\ i_2 + 1 &\le j_2 < n \\ i_1 &= j_2 - 1 \\ j_1 &= j_2 - i_2 - 1 \end{aligned} \right.$$

Delinearize the array accesses

Linearized one-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    A[i * n + j] =
        A[(n * j - n + j - i - 1];
```

$$\left\{ \text{null} \quad \begin{array}{r} 0 \leq i_1 < n \\ i_1 + 1 \leq j_1 < n \\ 0 \leq i_2 < n \\ i_2 + 1 \leq j_2 < n \\ i_1 \times n + j_1 = n \times j_2 - n + j_2 - i_2 - 1 \end{array} \right.$$

Delinearized multi-dimensional array

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j ++)
    B[i][j] = B[j - 1][j - i - 1];
```

$$\left\{ \text{null} \quad \begin{array}{r} 0 \leq i_1 < n \\ i_1 + 1 \leq j_1 < n \\ 0 \leq i_2 < n \\ i_2 + 1 \leq j_2 < n \\ i_1 = j_2 - 1 \\ j_1 = j_2 - i_2 - 1 \end{array} \right.$$

1. There is no integer solution, therefore, no dependence
2. The problem of delinearization requires to do QE over $\mathbb{Z}$ for non-linear expressions, which is, in principle, unfeasible. But see next slide.

Another approach to the delinearization problem

Principles

1. Assume that the delinearization problem has been solved for a particular problem instance, say 2D-Jacobi.

Another approach to the delinearization problem

Principles

1. Assume that the delinearization problem has been solved for a particular problem instance, say 2D-Jacobi.
2. Assume that we have another problem instance which looks very similar

Another approach to the delinearization problem

Principles

1. Assume that the delinearization problem has been solved for a particular problem instance, say 2D-Jacobi.
2. Assume that we have another problem instance which looks very similar
3. We may want to check whether the solved problem instance is obtained from the unsolved problem instance via a rank-preserving unimodular transformation between the two iteration domains.

Another approach to the delinearization problem

Principles

1. Assume that the delinearization problem has been solved for a particular problem instance, say 2D-Jacobi.
2. Assume that we have another problem instance which looks very similar
3. We may want to check whether the solved problem instance is obtained from the unsolved problem instance via a rank-preserving unimodular transformation between the two iteration domains.

Details

1. rank-preserving guarantees that the same array coefficients are read/written in the same order.

Another approach to the delinearization problem

Principles

1. Assume that the delinearization problem has been solved for a particular problem instance, say 2D-Jacobi.
2. Assume that we have another problem instance which looks very similar
3. We may want to check whether the solved problem instance is obtained from the unsolved problem instance via a rank-preserving unimodular transformation between the two iteration domains.

Details

1. rank-preserving guarantees that the same array coefficients are read/written in the same order.
2. rank-preserving transformations are "classifiable" off-line, next slide.

Another approach to the delinearization problem

Principles

1. Assume that the delinearization problem has been solved for a particular problem instance, say 2D-Jacobi.
2. Assume that we have another problem instance which looks very similar
3. We may want to check whether the solved problem instance is obtained from the unsolved problem instance via a rank-preserving unimodular transformation between the two iteration domains.

Details

1. rank-preserving guarantees that the same array coefficients are read/written in the same order.
2. rank-preserving transformations are "classifiable" off-line, next slide.
3. unimodularity guarantees that we can map integers to integers back and forth.

Another approach to the delinearization problem

Principles

1. Assume that the delinearization problem has been solved for a particular problem instance, say 2D-Jacobi.
2. Assume that we have another problem instance which looks very similar
3. We may want to check whether the solved problem instance is obtained from the unsolved problem instance via a rank-preserving unimodular transformation between the two iteration domains.

Details

1. rank-preserving guarantees that the same array coefficients are read/written in the same order.
2. rank-preserving transformations are "classifiable" off-line, next slide.
3. unimodularity guarantees that we can map integers to integers back and forth.
4. This can be performed at compile time (at the simple cost of linear algebra) and leads to a case discussion which can be evaluated at execution time.

3D Pattern matching problem

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & l \end{bmatrix} \times \begin{bmatrix} i \\ j \\ k \end{bmatrix} = \begin{bmatrix} ai + bj + ck \\ di + ej + fk \\ gi + hj + lf \end{bmatrix}$$

QE input:

$$\begin{aligned}
&\forall [i_1, j_1, k_1, i_2, j_2, k_2], \\
&(i_1 < i_2) \vee ((i_1 = i_2) \wedge (j_1 < j_2)) \vee ((i_1 = i_2) \wedge (j_1 = j_2) \wedge (k_1 < k_2)) \implies \\
&\quad (a\, i_1 + b\, j_1 + c\, k_1 < a\, i_2 + b\, j_2 + c\, k_2) \\
&\quad \vee((a\, i_1 + b\, j_1 + c\, k_1 = a\, i_2 + b\, j_2 + c\, k_2) \wedge (d\, i_1 + e\, j_1 + f\, k_1 < d\, i_2 + e\, j_2 + f\, k_2)) \\
&\quad \vee((a\, i_1 + b\, j_1 + c\, k_1 = a\, i_2 + b\, j_2 + c\, k_2) \wedge (d\, i_1 + e\, j_1 + f\, k_1 = d\, i_2 + e\, j_2 + f\, k_2) \\
&\qquad \wedge(g\, i_1 + h\, j_1 + l\, k_1 < g\, i_2 + h\, j_2 + l\, k_2))
\end{aligned}$$

QE output:

$$(f = 0) \wedge (0 < e) \wedge (c = 0) \wedge (b = 0) \wedge (0 < a) \wedge (0 < l)$$

which gives us the final matrix as below

$$\begin{bmatrix} a > 0 & 0 & 0 \\ c & e > 0 & 0 \\ g & h & l > 0 \end{bmatrix}$$

Plan

1. Overview
2. Basic concepts
3. Solving systems of linear inequalities
4. Integer hulls of polyhedra
5. Integer point counting for parametric polyhedra
6. Quantifier elimination over the integers
7. Concluding remarks

Plan

1. Overview
2. Basic concepts
2.1 Linear, affine, convex and conical hulls
2.2 Polyhedral sets
2.3 Farkas–Minkowsi–Weyl theorem
3. Solving systems of linear inequalities
3.1 Efficient removal of redundant inequalities
3.2 Implementation techniques
3.3 Experimentation and complexity estimates
4. Integer hulls of polyhedra
4.1 Motivations
4.2 Integer hulls, lattices and $\mathbb{Z}$-polyhedra
4.3 An integer hull algorithm
5. Integer point counting for parametric polyhedra
5.1 Motivations and objectives
5.2 Generating functions of non-parametric polyhedral sets
5.3 Integer point counting for parametric polyhedra
6. Quantifier elimination over the integers
6.1 Presburger arithmetic
6.2 Integer projection and quantifier elimination
7. Concluding remarks

1. This section is a review of the theory of polyhedral sets.

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

Notations 1

1. *As usual, we denote by $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ the ring of integers, the field of rational numbers, and the field of real numbers.*

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

Notations 1

1. *As usual, we denote by* $\mathbb{Z}$, $\mathbb{Q}$, *and* $\mathbb{R}$ *the ring of integers, the field of rational numbers, and the field of real numbers.*
2. *We consider the* d*-dimensional Euclidean space* $\mathbb{R}^d$ *equipped with the Euclidean topology.*

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

Notations 1

1. *As usual, we denote by* $\mathbb{Z}$, $\mathbb{Q}$, *and* $\mathbb{R}$ *the ring of integers, the field of rational numbers, and the field of real numbers.*
2. *We consider the* d*-dimensional Euclidean space* $\mathbb{R}^d$ *equipped with the Euclidean topology.*
3. *Upper case letters* $A, B, \ldots, X, \ldots$ *will usually denote matrices or polyhedra, sometimes polynomials.*

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

Notations 1

1. *As usual, we denote by* $\mathbb{Z}$*,* $\mathbb{Q}$*, and* $\mathbb{R}$ *the ring of integers, the field of rational numbers, and the field of real numbers.*
2. *We consider the* d*-dimensional Euclidean space* $\mathbb{R}^d$ *equipped with the Euclidean topology.*
3. *Upper case letters* $A, B, \ldots, X, \ldots$ *will usually denote matrices or polyhedra, sometimes polynomials.*
4. *Bold lower case letters* $\mathbf{a}, \mathbf{b}, \ldots, \mathbf{x}, \ldots$ *will usually denote vectors or points.*

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

Notations 1

1. *As usual, we denote by* $\mathbb{Z}$*,* $\mathbb{Q}$*, and* $\mathbb{R}$ *the ring of integers, the field of rational numbers, and the field of real numbers.*
2. *We consider the* d*-dimensional Euclidean space* $\mathbb{R}^d$ *equipped with the Euclidean topology.*
3. *Upper case letters* $A, B, \ldots, X, \ldots$ *will usually denote matrices or polyhedra, sometimes polynomials.*
4. *Bold lower case letters* $\mathbf{a}, \mathbf{b}, \ldots, \mathbf{x}, \ldots$ *will usually denote vectors or points.*
5. *Non-bold case letters* $a, b, \ldots, x \ldots$ *will usually denote scalars or points (outside the context of linear algebra).*

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

Notations 1

1. *As usual, we denote by* $\mathbb{Z}$, $\mathbb{Q}$, *and* $\mathbb{R}$ *the ring of integers, the field of rational numbers, and the field of real numbers.*
2. *We consider the* d*-dimensional Euclidean space* $\mathbb{R}^d$ *equipped with the Euclidean topology.*
3. *Upper case letters* $A, B, \ldots, X, \ldots$ *will usually denote matrices or polyhedra, sometimes polynomials.*
4. *Bold lower case letters* $\mathbf{a}, \mathbf{b}, \ldots, \mathbf{x}, \ldots$ *will usually denote vectors or points.*
5. *Non-bold case letters* $a, b, \ldots, x \ldots$ *will usually denote scalars or points (outside the context of linear algebra).*
6. *Let* K *be a subset of* $\mathbb{R}^d$*.*

1. This section is a review of the theory of polyhedral sets.
2. It is based on the books of Branko Grünbaum [6] and Alexander Schrijver [18], where the missing proofs can be found.

Notations 1

1. *As usual, we denote by* $\mathbb{Z}$, $\mathbb{Q}$, *and* $\mathbb{R}$ *the ring of integers, the field of rational numbers, and the field of real numbers.*
2. *We consider the* d*-dimensional Euclidean space* $\mathbb{R}^d$ *equipped with the Euclidean topology.*
3. *Upper case letters* $A, B, \ldots, X, \ldots$ *will usually denote matrices or polyhedra, sometimes polynomials.*
4. *Bold lower case letters* $\mathbf{a}, \mathbf{b}, \ldots, \mathbf{x}, \ldots$ *will usually denote vectors or points.*
5. *Non-bold case letters* $a, b, \ldots, x \ldots$ *will usually denote scalars or points (outside the context of linear algebra).*
6. *Let* K *be a subset of* $\mathbb{R}^d$*.*
7. *We denote by* $\overline{K}, \mathring{K}, \partial K$ *the closure, the interior the border of* K*.*

Plan

Affine, conical and convex hulls

Definition 1

Let $X \subseteq \mathbb{R}^d$. The **span**, the **affine hull**, the **convex hull**, the **conical hull** of X, denoted $\mathrm{Span}(X)$, $\mathrm{AffineHull}(X)$, $\mathrm{ConvexHull}(X)$, $\mathrm{ConicalHull}(X)$, are defined by: For every $y \in \mathbb{R}^d$, we have:

1. $y \in \mathrm{Span}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.

Affine, conical and convex hulls

Definition 1

Let $X \subseteq \mathbb{R}^d$. The **span**, the **affine hull**, the **convex hull**, the **conical hull** of X, denoted $\text{Span}(X)$, $\text{AffineHull}(X)$, $\text{ConvexHull}(X)$, $\text{ConicalHull}(X)$, are defined by: For every $y \in \mathbb{R}^d$, we have:

1. $y \in \text{Span}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.
2. $y \in \text{AffineHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$ and $\lambda_1 + \cdots + \lambda_e = 1$.

Affine, conical and convex hulls

Definition 1

Let $X \subseteq \mathbb{R}^d$. The **span**, the **affine hull**, the **convex hull**, the **conical hull** of X, denoted $\mathrm{Span}(X)$, $\mathrm{AffineHull}(X)$, $\mathrm{ConvexHull}(X)$, $\mathrm{ConicalHull}(X)$, are defined by: For every $y \in \mathbb{R}^d$, we have:

1. $y \in \mathrm{Span}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e \mid y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.
2. $y \in \mathrm{AffineHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e \mid y = \lambda_1 x_1 + \cdots + \lambda_e x_e \text{ and } \lambda_1 + \cdots + \lambda_e = 1$.
3. $y \in \mathrm{ConvexHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e \mid y = \lambda_1 x_1 + \cdots + \lambda_e x_e \text{ and } \lambda_1 + \cdots + \lambda_e = 1$.

Affine, conical and convex hulls

Definition 1

Let $X \subseteq \mathbb{R}^d$. The **span**, the **affine hull**, the **convex hull**, the **conical hull** of X, denoted Span(X), AffineHull(X), ConvexHull(X), ConicalHull(X), are defined by: For every $y \in \mathbb{R}^d$, we have:

1. $y \in \text{Span}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.
2. $y \in \text{AffineHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$ and $\lambda_1 + \cdots + \lambda_e = 1$.
3. $y \in \text{ConvexHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$ and $\lambda_1 + \cdots + \lambda_e = 1$.
4. $y \in \text{ConicalHull}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.

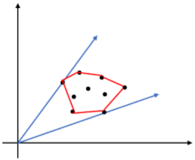
Affine, conical and convex hulls

Definition 1

Let $X \subseteq \mathbb{R}^d$. The **span**, the **affine hull**, the **convex hull**, the **conical hull** of X, denoted $\mathrm{Span}(X)$, $\mathrm{AffineHull}(X)$, $\mathrm{ConvexHull}(X)$, $\mathrm{ConicalHull}(X)$, are defined by: For every $y \in \mathbb{R}^d$, we have:

1. $y \in \mathrm{Span}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e | \; y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.
2. $y \in \mathrm{AffineHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e | \; y = \lambda_1 x_1 + \cdots + \lambda_e x_e \text{ and } \lambda_1 + \cdots + \lambda_e = 1$.
3. $y \in \mathrm{ConvexHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e | \; y = \lambda_1 x_1 + \cdots + \lambda_e x_e \text{ and } \lambda_1 + \cdots + \lambda_e = 1$.
4. $y \in \mathrm{ConicalHull}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e | \; y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.

In the plane, the **conical** and **convex** hulls of a few points.



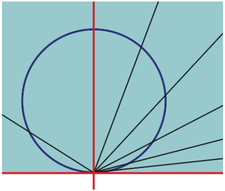
Affine, conical and convex hulls

Definition 2

Let $X \subseteq \mathbb{R}^d$. The **span**, the **affine hull**, the **convex hull**, the **conical hull** of X, denoted Span(X), AffineHull(X), ConvexHull(X), ConicalHull(X), are defined by: For every $y \in \mathbb{R}^d$, we have:

1. $y \in \text{Span}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.
2. $y \in \text{AffineHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$ and $\lambda_1 + \cdots + \lambda_e = 1$.
3. $y \in \text{ConvexHull}(X) \iff \exists e \in \mathbb{N}_{>0}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$ and $\lambda_1 + \cdots + \lambda_e = 1$.
4. $y \in \text{ConicalHull}(X) \iff \exists e \in \mathbb{N}, \exists (x_1, \ldots, x_e) \in X^e, \exists (\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e |$ $y = \lambda_1 x_1 + \cdots + \lambda_e x_e$.

In the plane, the **conical** hull of a **circle** passing through the origin is the open half-plane defined by the tangent line to the circle at the origin plus the origin.



Some properties of hulls and spans (1/3)

Definition 3

For $X, Y \subseteq \mathbb{R}^d$ and $x_0 \in \mathbb{R}^d$, we define the **Minkowski sum** of X and Y as

$$X + Y \quad = \quad \{x + y \mid x \in X \mathrm{and} y \in Y\},$$

and we write $X + x_0$ for $X + \{x_0\}$.

Some properties of hulls and spans (1/3)

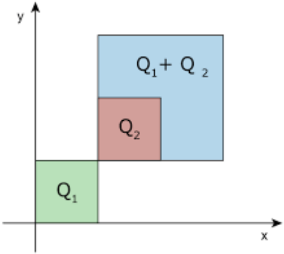
Definition 3

For $X, Y \subseteq \mathbb{R}^d$ and $x_0 \in \mathbb{R}^d$, we define the **Minkowski sum** of X and Y as

$$X + Y = \{x + y \mid x \in X \mathrm{and} y \in Y\},$$

and we write $X + x_0$ for $X + \{x_0\}$.

In the plane, the blue polyhedron is the Minkowski sum of the red and green polyhedra.



Some properties of hulls and spans (1/3)

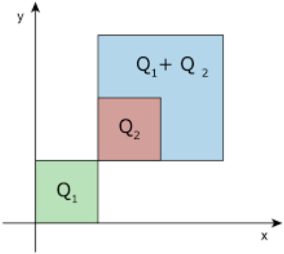
Definition 3

For $X, Y \subseteq \mathbb{R}^d$ and $x_0 \in \mathbb{R}^d$, we define the **Minkowski sum** of X and Y as

$$X + Y = \{x + y \mid x \in X \text{and} y \in Y\},$$

and we write $X + x_0$ for $X + \{x_0\}$.

In the plane, the blue polyhedron is the Minkowski sum of the red and green polyhedra.



Definition 4

A subset $P \subseteq \mathbb{R}^d$ is a **polytope**, if it is the convex hull of finitely many points, that is, if there exists a finite set $X \subseteq \mathbb{R}^d$ so that $P = \text{ConvexHull}(X)$.

Some properties of hulls and spans (1/3)

Definition 5

For $X, Y \subseteq \mathbb{R}^d$ and $x_0 \in \mathbb{R}^d$, we define the **Minkowski sum** of X and Y as

$$X + Y \quad = \quad \{x + y \mid x \in X \mathrm{and} y \in Y\},$$

and we write $X + x_0$ for $X + \{x_0\}$.

Proposition 1

If $x_0 \in X$, then we have $\text{AffineHull}(X) = x_0 + \text{Span}(X)$.

Proof.

□

Some properties of hulls and spans (1/3)

Definition 5

For $X, Y \subseteq \mathbb{R}^d$ and $x_0 \in \mathbb{R}^d$, we define the **Minkowski sum** of X and Y as

$$X + Y \quad = \quad \{x + y \mid x \in X \mathrm{and} y \in Y\},$$

and we write $X + x_0$ for $X + \{x_0\}$.

Proposition 1

If $x_0 \in X$, then we have $\mathrm{AffineHull}(X) = x_0 + \mathrm{Span}(X)$.

Proof.

Assume $x_0 \in X$ and let $x \in \mathrm{AffineHull}(X)$. Then, there exists $(x_1, \ldots, x_e) \in X^e$ and $(\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e$ such that $x = \lambda_1 x_1 + \cdots + \lambda_e x_e$ and $\lambda_1 + \cdots + \lambda_e = 1$. Then, we have:

$$x = x_0 + \lambda_1(x_1 - x_0) + \cdots \lambda_e(x_e - x_0),$$

that is, $x \in x_0 + \mathrm{Span}(X)$.

□

Some properties of hulls and spans (1/3)

Definition 5

For $X, Y \subseteq \mathbb{R}^d$ and $x_0 \in \mathbb{R}^d$, we define the **Minkowski sum** of X and Y as

$$X + Y \quad = \quad \{x + y \mid x \in X \mathrm{and} y \in Y\},$$

and we write $X + x_0$ for $X + \{x_0\}$.

Proposition 1

If $x_0 \in X$, then we have AffineHull$(X) = x_0 +$ Span(X).

Proof.

Assume $x_0 \in X$ and let $x \in$ AffineHull(X). Then, there exists $(x_1, \ldots, x_e) \in X^e$ and $(\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e$ such that $x = \lambda_1 x_1 + \cdots + \lambda_e x_e$ and $\lambda_1 + \cdots + \lambda_e = 1$. Then, we have:

$$x = x_0 + \lambda_1(x_1 - x_0) + \cdots \lambda_e(x_e - x_0),$$

that is, $x \in x_0 +$ Span(X). Conversely, let $x \in\in x_0 +$ Span(X). Let $(\lambda_1, \ldots, \lambda_e) \in \mathbb{R}^e$ such that $x = x_0 + \lambda_1 x_1 + \cdots \lambda_e x_e$. Define $\mu_i = \lambda_i$, for $1 \leq i \leq e$ and $\mu_0 = 1 - \lambda_1 - \cdots - \lambda_e$. Then, we have:

$$x = \mu_0 x_0 + \mu_1 x_1 + \cdots + \mu_e x_e,$$

that is, $x \in$ AffineHull(X). □

Some properties of hulls and spans (2/3)

Proposition 2

We have:

$$\text{ConicalHull}(X) \;=\; \{x \in \mathbb{R}^d \mid \exists t \in \mathbb{R}_{>0}\ tx \in \text{ConvexHull}(X)\} \cup \{0\}.$$

Some properties of hulls and spans (2/3)

Proposition 2

We have:

$$\text{ConicalHull}(X) \;=\; \{x \in \mathbb{R}^d \mid \exists t \in \mathbb{R}_{>0}\ tx \in \text{ConvexHull}(X)\} \cup \{0\}.$$

Proof.

Denote by Z the set on the right-hand side of the equality. Let $x \in Z$. If $x = 0$ holds then $x \in \text{ConicalHull}(X)$ holds. Assume from now on that $x \neq 0$. Let $t \in \mathbb{R}_{>0}$ such that $tx \in \text{ConvexHull}(X)$ holds. Then, let $e \in \mathbb{N}_{>0}$, let $(x_1, \ldots, x_e) \in X^e$, let $(\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e$ such that

$$tx = \lambda_1 x_1 + \cdots + \lambda_e x_e \text{ and } \lambda_1 + \cdots + \lambda_e = 1.$$

It follows that $x \in \text{ConicalHull}(X)$ holds.

□

Some properties of hulls and spans (2/3)

Proposition 2

We have:

$$\text{ConicalHull}(X) \;=\; \{x \in \mathbb{R}^d \mid \exists t \in \mathbb{R}_{>0} \; tx \in \text{ConvexHull}(X)\} \cup \{0\}.$$

Proof.

Denote by Z the set on the right-hand side of the equality. Let $x \in Z$. If $x = 0$ holds then $x \in \text{ConicalHull}(X)$ holds. Assume from now on that $x \neq 0$. Let $t \in \mathbb{R}_{>0}$ such that $tx \in \text{ConvexHull}(X)$ holds. Then, let $e \in \mathbb{N}_{>0}$, let $(x_1, \ldots, x_e) \in X^e$, let $(\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e$ such that

$$tx = \lambda_1 x_1 + \cdots + \lambda_e x_e \text{ and } \lambda_1 + \cdots + \lambda_e = 1.$$

It follows that $x \in \text{ConicalHull}(X)$ holds. Conversely, let $x \in \text{ConicalHull}(X)$. If $x = 0$ holds then $x \in Z$ holds. Assume from now on that $x \neq 0$. Then, let $e \in \mathbb{N}_{>0}$, let $(x_1, \ldots, x_e) \in X^e$, let $(\lambda_1, \ldots, \lambda_e) \in {\mathbb{R}_{\geq 0}}^e$ such that

$$x = \lambda_1 x_1 + \cdots + \lambda_e x_e \text{ holds.}$$

Since $x \neq 0$, we have $\lambda := \lambda_1 + \cdots + \lambda_e \neq 0$. It follows that $\frac{x}{\lambda} \in X$, that is, $x \in Z$. □

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

1. $\mathrm{ConvexHull}(X + Y) = \mathrm{ConvexHull}(X) + \mathrm{ConvexHull}(Y)$ *holds,*

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

1. $\mathrm{ConvexHull}(X + Y) = \mathrm{ConvexHull}(X) + \mathrm{ConvexHull}(Y)$ *holds,*
2. $X \subseteq \mathrm{ConvexHull}(X)$ *holds and* $\mathrm{ConvexHull}(X)$ *is convex.*

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

1. $\text{ConvexHull}(X + Y) = \text{ConvexHull}(X) + \text{ConvexHull}(Y)$ *holds,*
2. $X \subseteq \text{ConvexHull}(X)$ *holds and* $\text{ConvexHull}(X)$ *is convex.*
3. $\text{ConvexHull}(X)$ *contains any convex set that contains* X*.*

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

1. $\mathrm{ConvexHull}(X + Y) = \mathrm{ConvexHull}(X) + \mathrm{ConvexHull}(Y)$ *holds,*
2. $X \subseteq \mathrm{ConvexHull}(X)$ *holds and* $\mathrm{ConvexHull}(X)$ *is convex.*
3. $\mathrm{ConvexHull}(X)$ *contains any convex set that contains* X*.*
4. $X \subseteq \mathrm{ConicalHull}(X)$ *holds and* $\mathrm{ConicalHull}(X)$ *is convex.*

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

1. $\mathrm{ConvexHull}(X + Y) = \mathrm{ConvexHull}(X) + \mathrm{ConvexHull}(Y)$ *holds,*
2. $X \subseteq \mathrm{ConvexHull}(X)$ *holds and* $\mathrm{ConvexHull}(X)$ *is convex.*
3. $\mathrm{ConvexHull}(X)$ *contains any convex set that contains* X.
4. $X \subseteq \mathrm{ConicalHull}(X)$ *holds and* $\mathrm{ConicalHull}(X)$ *is convex.*
5. $\mathrm{ConicalHull}(X)$ *is a* **convex cone**, *that is: it contains the origin, it is closed under addition and multiplication by a non-negative scalar.*

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

1. $\mathrm{ConvexHull}(X + Y) = \mathrm{ConvexHull}(X) + \mathrm{ConvexHull}(Y)$ *holds,*
2. $X \subseteq \mathrm{ConvexHull}(X)$ *holds and* $\mathrm{ConvexHull}(X)$ *is convex.*
3. $\mathrm{ConvexHull}(X)$ *contains any convex set that contains* X*.*
4. $X \subseteq \mathrm{ConicalHull}(X)$ *holds and* $\mathrm{ConicalHull}(X)$ *is convex.*
5. $\mathrm{ConicalHull}(X)$ *is a* **convex cone***, that is: it contains the origin, it is closed under addition and multiplication by a non-negative scalar.*

Proof is routine.

Some properties of hulls and spans (3/3)

Proposition 3

For all $X, Y \subseteq \mathbb{R}^d$, the following properties hold:

1. $\text{ConvexHull}(X + Y) = \text{ConvexHull}(X) + \text{ConvexHull}(Y)$ *holds,*
2. $X \subseteq \text{ConvexHull}(X)$ *holds and* $\text{ConvexHull}(X)$ *is convex.*
3. $\text{ConvexHull}(X)$ *contains any convex set that contains* X*.*
4. $X \subseteq \text{ConicalHull}(X)$ *holds and* $\text{ConicalHull}(X)$ *is convex.*
5. $\text{ConicalHull}(X)$ *is a* **convex cone***, that is: it contains the origin, it is closed under addition and multiplication by a non-negative scalar.*

Proof is routine.

Definition 6

A finite set $X \subseteq \mathbb{R}^d$ is called **affinely independent** if for every $x \in X$ we have $x \notin \text{AffineHull}(X \setminus \{x\})$, that is, $\{y - x \mid y \in X \text{ and } y \neq x\}$ is linearly independent for each $x \in X$.

Supporting hyperplanes

Notations 2

Let $\alpha \in \mathbb{R}^d$*, let* $\beta \in \mathbb{R}$ *and denote by* H *the hyperplane defined by*

$$H = \{x \in \mathbb{R}^d \mid \alpha^T x = \beta\}.$$

Supporting hyperplanes

Notations 2

Let $\alpha \in \mathbb{R}^d$*, let* $\beta \in \mathbb{R}$ *and denote by* H *the hyperplane defined by*

$$H = \{x \in \mathbb{R}^d \mid \alpha^T x = \beta\}.$$

Definition 7

We say that the hyperplane H **supports** K if either

$$\sup\{\alpha^T x \mid x \in K\} = \beta, \quad \text{or} \quad \inf\{\alpha^T x \mid x \in K\} = \beta$$

holds, but not both.

Supporting hyperplanes

Notations 2

Let $\alpha \in \mathbb{R}^d$*, let* $\beta \in \mathbb{R}$ *and denote by* H *the hyperplane defined by*

$$H = \{x \in \mathbb{R}^d \mid \alpha^T x = \beta\}.$$

Definition 7

We say that the hyperplane H **supports** K if either

$$\sup\{\alpha^T x \mid x \in K\} = \beta, \quad \text{or} \quad \inf\{\alpha^T x \mid x \in K\} = \beta$$

holds, but not both.

Faces

Definition 8

1. A set $F \subseteq K$ is a **face** if either $F = \emptyset$, or $F = K$, or if there exists a hyperplane H supporting K such that we have $F = K \cap H$.

Faces

Definition 8

1. A set $F \subseteq K$ is a **face** if either $F = \emptyset$, or $F = K$, or if there exists a hyperplane H supporting K such that we have $F = K \cap H$.
2. $\text{Faces}(K)$ denotes the set of all faces of K.

Faces

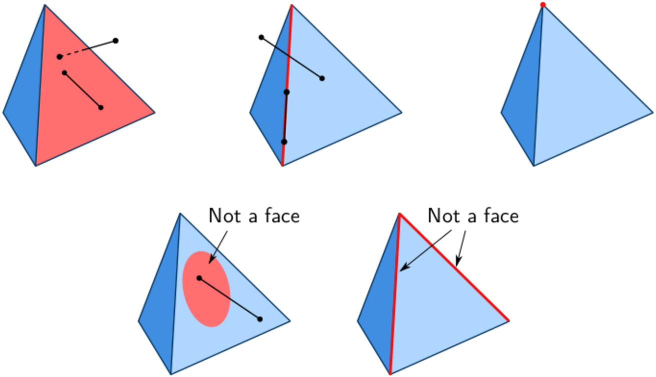
Definition 8

1. A set $F \subseteq K$ is a **face** if either $F = \emptyset$, or $F = K$, or if there exists a hyperplane H supporting K such that we have $F = K \cap H$.
2. $\text{Faces}(K)$ denotes the set of all faces of K.
3. We say $F \in \text{Faces}(K)$ is **proper** if $F \neq \emptyset$ and $F \neq K$.

Faces

Definition 8

1. A set $F \subseteq K$ is a **face** if either $F = \emptyset$, or $F = K$, or if there exists a hyperplane H supporting K such that we have $F = K \cap H$.
2. $\text{Faces}(K)$ denotes the set of all faces of K.
3. We say $F \in \text{Faces}(K)$ is **proper** if $F \neq \emptyset$ and $F \neq K$.



Extreme points and exposed points

Notations 3

From now on, let us assume that $K \subseteq \mathbb{R}^d$ is convex.

Extreme points and exposed points

Notations 3

From now on, let us assume that $K \subseteq \mathbb{R}^d$ is convex.

Definition 9

1. A point $x \in K$ is an **extreme point** of K if it does not belong to the relative interior of any segment contained in K.

Extreme points and exposed points

Notations 3

From now on, let us assume that $K \subseteq \mathbb{R}^d$ is convex.

Definition 9

1. A point $x \in K$ is an **extreme point** of K if it does not belong to the relative interior of any segment contained in K.
2. ExtremePoints(K) denotes the set of all extreme points of K.

Extreme points and exposed points

Notations 3

From now on, let us assume that $K \subseteq \mathbb{R}^d$ is convex.

Definition 9

1. A point $x \in K$ is an **extreme point** of K if it does not belong to the relative interior of any segment contained in K.
2. ExtremePoints(K) denotes the set of all extreme points of K.
3. A point $x \in K$ is an **exposed point** of K if $\{x\} \in$ Faces(K) holds.

Extreme points and exposed points

Notations 3

From now on, let us assume that $K \subseteq \mathbb{R}^d$ is convex.

Definition 9

1. A point $x \in K$ is an **extreme point** of K if it does not belong to the relative interior of any segment contained in K.
2. ExtremePoints(K) denotes the set of all extreme points of K.
3. A point $x \in K$ is an **exposed point** of K if $\{x\} \in$ Faces(K) holds.
4. ExposedPoints(K) denotes the set of exposed points of K.

Extreme points and exposed points

Notations 3

From now on, let us assume that $K \subseteq \mathbb{R}^d$ is convex.

Definition 9

1. A point $x \in K$ is an **extreme point** of K if it does not belong to the relative interior of any segment contained in K.
2. ExtremePoints(K) denotes the set of all extreme points of K.
3. A point $x \in K$ is an **exposed point** of K if $\{x\} \in$ Faces(K) holds.
4. ExposedPoints(K) denotes the set of exposed points of K.

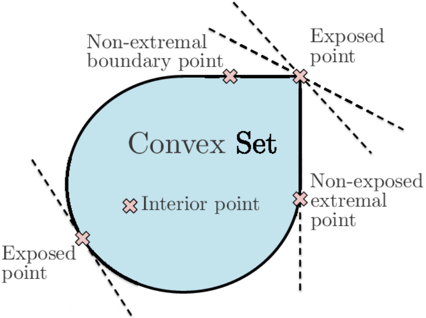
Extreme points and exposed points

Notations 3

From now on, let us assume that $K \subseteq \mathbb{R}^d$ is convex.

Definition 9

1. A point $x \in K$ is an **extreme point** of K if it does not belong to the relative interior of any segment contained in K.
2. ExtremePoints(K) denotes the set of all extreme points of K.
3. A point $x \in K$ is an **exposed point** of K if $\{x\} \in$ Faces(K) holds.
4. ExposedPoints(K) denotes the set of exposed points of K.



Some properties of faces, exposed and extreme points

Proposition 4

The following properties hold.

1. *For all* $F \in \text{Faces}(K)$*, we have:*

$$\text{ExtremePoints}(F) = F \cap \text{ExtremePoints}(K).$$

Some properties of faces, exposed and extreme points

Proposition 4

The following properties hold.

1. *For all* $F \in \text{Faces}(K)$, *we have:*

$$\text{ExtremePoints}(F) = F \cap \text{ExtremePoints}(K).$$

2. *We have:*

$$\text{ExposedPoints}(K) \subseteq \text{ExtremePoints}(K).$$

Some properties of faces, exposed and extreme points

Proposition 4

The following properties hold.

1. *For all* $F \in \mathrm{Faces}(K)$, *we have:*
$$\mathrm{ExtremePoints}(F) = F \cap \mathrm{ExtremePoints}(K).$$
2. *We have:*
$$\mathrm{ExposedPoints}(K) \subseteq \mathrm{ExtremePoints}(K).$$
3. *If* K *is compact then:*
$$\overline{\mathrm{ConvexHull}(\mathrm{ExtremePoints}(K))} = K.$$

Some properties of faces, exposed and extreme points

Proposition 4

The following properties hold.

1. *For all* $F \in \mathrm{Faces}(K)$*, we have:*
$$\mathrm{ExtremePoints}(F) = F \cap \mathrm{ExtremePoints}(K).$$
2. *We have:*
$$\mathrm{ExposedPoints}(K) \subseteq \mathrm{ExtremePoints}(K).$$
3. *If* K *is compact then:*
$$\overline{\mathrm{ConvexHull}(\mathrm{ExtremePoints}(K))} = K.$$
4. *If* K *is closed then:*
$$\mathrm{ExtremePoints}(K) \subseteq \overline{\mathrm{ExposedPoints}(K)}.$$

Some properties of faces, exposed and extreme points

Proposition 4

The following properties hold.

1. *For all* $F \in \mathrm{Faces}(K)$*, we have:*
$$\mathrm{ExtremePoints}(F) = F \cap \mathrm{ExtremePoints}(K).$$
2. *We have:*
$$\mathrm{ExposedPoints}(K) \subseteq \mathrm{ExtremePoints}(K).$$
3. *If* K *is compact then:*
$$\overline{\mathrm{ConvexHull}(\mathrm{ExtremePoints}(K))} = K.$$
4. *If* K *is closed then:*
$$\mathrm{ExtremePoints}(K) \subseteq \overline{\mathrm{ExposedPoints}(K)}.$$
5. *The intersection of any family of faces of* K *is itself a face of* K*.*

Some properties of faces, exposed and extreme points

Proposition 4

The following properties hold.

1. *For all* $F \in \text{Faces}(K)$*, we have:*
$$\text{ExtremePoints}(F) = F \cap \text{ExtremePoints}(K).$$
2. *We have:*
$$\text{ExposedPoints}(K) \subseteq \text{ExtremePoints}(K).$$
3. *If* K *is compact then:*
$$\overline{\text{ConvexHull}(\text{ExtremePoints}(K))} = K.$$
4. *If* K *is closed then:*
$$\text{ExtremePoints}(K) \subseteq \overline{\text{ExposedPoints}(K)}.$$
5. *The intersection of any family of faces of* K *is itself a face of* K*.*
6. K *is unbounded if and only if it contains a ray (= half-line).*

Some properties of faces, exposed and extreme points

Proposition 4

The following properties hold.

1. *For all* $F \in \text{Faces}(K)$, *we have:*
$$\text{ExtremePoints}(F) = F \cap \text{ExtremePoints}(K).$$
2. *We have:*
$$\text{ExposedPoints}(K) \subseteq \text{ExtremePoints}(K).$$
3. *If* K *is compact then:*
$$\overline{\text{ConvexHull}(\text{ExtremePoints}(K))} = K.$$
4. *If* K *is closed then:*
$$\text{ExtremePoints}(K) \subseteq \overline{\text{ExposedPoints}(K)}.$$
5. *The intersection of any family of faces of* K *is itself a face of* K.
6. K *is unbounded if and only if it contains a ray (= half-line).*
7. *Assume* K *is closed. let* $L = \{\lambda z \mid \lambda \geq 0\}$ *be a ray emanating from the origin and let* $x, y \in K$. *Then, we have:*
$$x + L \subseteq K \iff y + L \subseteq K.$$

Cones

Definition 10

1. A non-empty set $C \subseteq \mathbb{R}^d$ is a **cone** if: $\forall \lambda \in \mathbb{R}_{\geq 0}\ \lambda \mathbf{x} \in C$.

Cones

Definition 10

1. A non-empty set $C \subseteq \mathbb{R}^d$ is a **cone** if: $\forall \lambda \in \mathbb{R}_{\geq 0}\ \lambda \mathbf{x} \in C$.
2. A non-empty set $C \subseteq \mathbb{R}^d$ is a **convex cone** if:

$$\forall \lambda, \mu \in \mathbb{R}_{\geq 0} \quad \forall \mathbf{x}, \mathbf{y} \in C \quad \lambda \mathbf{x} + \mu \mathbf{y} \in C.$$

Cones

Definition 10

1. A non-empty set $C \subseteq \mathbb{R}^d$ is a **cone** if: $\forall \lambda \in \mathbb{R}_{\geq 0}\ \lambda \mathbf{x} \in C$.
2. A non-empty set $C \subseteq \mathbb{R}^d$ is a **convex cone** if:
$$\forall \lambda, \mu \in \mathbb{R}_{\geq 0} \quad \forall \mathbf{x}, \mathbf{y} \in C \quad \lambda \mathbf{x} + \mu \mathbf{y} \in C.$$
3. The cone $C \subseteq \mathbb{R}^d$ is **polyhedral**, if, for some matrix $A \in \mathbb{R}^{m \times d}$, with $m \in \mathbb{N}_{>0}$, we have:
$$C = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{0}\}.$$

Cones

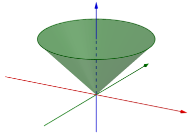
Definition 10

1. A non-empty set $C \subseteq \mathbb{R}^d$ is a **cone** if: $\forall \lambda \in \mathbb{R}_{\geq 0}\ \lambda \mathbf{x} \in C$.
2. A non-empty set $C \subseteq \mathbb{R}^d$ is a **convex cone** if:
$$\forall \lambda, \mu \in \mathbb{R}_{\geq 0} \quad \forall \mathbf{x}, \mathbf{y} \in C \quad \lambda \mathbf{x} + \mu \mathbf{y} \in C.$$
3. The cone $C \subseteq \mathbb{R}^d$ is **polyhedral**, if, for some matrix $A \in \mathbb{R}^{m \times d}$, with $m \in \mathbb{N}_{>0}$, we have:
$$C = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{0}\}.$$
4. The cone $C \subseteq \mathbb{R}^d$ is **finitely generated** by $\mathbf{x}_1, \ldots, \mathbf{x}_e \in \mathbb{R}^d$, if we have:
$$C = \mathrm{ConicalHull}(\{\mathbf{x}_1, \ldots, \mathbf{x}_e\}).$$

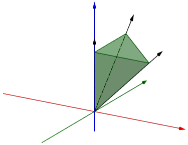
Cones

Definition 10

1. A non-empty set $C \subseteq \mathbb{R}^d$ is a **cone** if: $\forall \lambda \in \mathbb{R}_{\geq 0}\ \lambda \mathbf{x} \in C$.
2. A non-empty set $C \subseteq \mathbb{R}^d$ is a **convex cone** if:
$$\forall \lambda, \mu \in \mathbb{R}_{\geq 0} \quad \forall \mathbf{x}, \mathbf{y} \in C \quad \lambda \mathbf{x} + \mu \mathbf{y} \in C.$$
3. The cone $C \subseteq \mathbb{R}^d$ is **polyhedral**, if, for some matrix $A \in \mathbb{R}^{m \times d}$, with $m \in \mathbb{N}_{>0}$, we have:
$$C \;=\; \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{0}\}.$$
4. The cone $C \subseteq \mathbb{R}^d$ is **finitely generated** by $\mathbf{x}_1, \ldots, \mathbf{x}_e \in \mathbb{R}^d$, if we have:
$$C \;=\; \text{ConicalHull}(\{\mathbf{x}_1, \ldots, \mathbf{x}_e\}).$$



A non-polyhedral cone.



A polyhedral cone.

Dual cones

Definition 11

For a subset $C \subseteq \mathbb{R}^d$, the **dual cone** is given by:

$$C^* = \left\{ \mathbf{y} \in \mathbb{R}^d \mid \mathbf{y}^t \mathbf{x} \geq 0 \qquad \forall \mathbf{x} \in C \right\}.$$

Dual cones

Definition 11

For a subset $C \subseteq \mathbb{R}^d$, the **dual cone** is given by:

$$C^* = \left\{ \mathbf{y} \in \mathbb{R}^d \mid \mathbf{y}^t \mathbf{x} \geq 0 \qquad \forall \mathbf{x} \in C \right\}.$$

For a subset $C \subseteq \mathbb{R}^d$, the **polar cone** is given by:

$$C^0 = \left\{ \mathbf{y} \in \mathbb{R}^d \mid \mathbf{y}^t \mathbf{x} \leq 0 \qquad \forall \mathbf{x} \in C \right\},$$

Dual cones

Definition 11

For a subset $C \subseteq \mathbb{R}^d$, the **dual cone** is given by:

$$C^* = \left\{ \mathbf{y} \in \mathbb{R}^d \mid \mathbf{y}^t \mathbf{x} \geq 0 \qquad \forall \mathbf{x} \in C \right\}.$$

For a subset $C \subseteq \mathbb{R}^d$, the **polar cone** is given by:

$$C^0 = \left\{ \mathbf{y} \in \mathbb{R}^d \mid \mathbf{y}^t \mathbf{x} \leq 0 \qquad \forall \mathbf{x} \in C \right\},$$

Proposition 5

The dual cone C^* of $C \subseteq \mathbb{R}^d$ is a convex cone and we have $C^0 = -C^*$.

Dual cones

Definition 11

For a subset $C \subseteq \mathbb{R}^d$, the **dual cone** is given by:

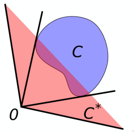
$$C^* = \left\{ \mathbf{y} \in \mathbb{R}^d \mid \mathbf{y}^t \mathbf{x} \geq 0 \quad \forall \mathbf{x} \in C \right\}.$$

For a subset $C \subseteq \mathbb{R}^d$, the **polar cone** is given by:

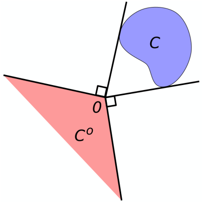
$$C^0 = \left\{ \mathbf{y} \in \mathbb{R}^d \mid \mathbf{y}^t \mathbf{x} \leq 0 \quad \forall \mathbf{x} \in C \right\},$$

Proposition 5

The dual cone C^* of $C \subseteq \mathbb{R}^d$ is a convex cone and we have $C^0 = -C^*$.



A cranberry and its dual.



A cranberry and its polar cone.

Polar cone theorem

Theorem 12

For a subset $C \subseteq \mathbb{R}^d$, we have:

$$C^{0^0} = \overline{\text{ConicalHull}(C)}.$$

In particular, if C is closed and convex, then we have:

$$C^{0^0} = C.$$

Polar cone theorem

Theorem 12

For a subset $C \subseteq \mathbb{R}^d$, we have:

$$C^{0^0} = \overline{\text{ConicalHull}(C)}.$$

In particular, if C is closed and convex, then we have:

$$C^{0^0} = C.$$

Proof when C is closed and convex.

Since we have $\mathbf{y}^t\mathbf{x} \leq 0$ for all $\mathbf{y} \in C^0$ and all $\mathbf{x} \in C$ it follows that we have $C \subseteq C^{0^0}$.

□

Polar cone theorem

Theorem 12

For a subset $C \subseteq \mathbb{R}^d$, we have:

$$C^{0^0} = \overline{\text{ConicalHull}(C)}.$$

In particular, if C is closed and convex, then we have:

$$C^{0^0} = C.$$

Proof when C is closed and convex.

Since we have $\mathbf{y}^t\mathbf{x} \leq 0$ for all $\mathbf{y} \in C^0$ and all $\mathbf{x} \in C$ it follows that we have $C \subseteq C^{0^0}$. To prove the reverse inclusion, take $\mathbf{z} \in C^{0^0}$. Let $\hat{\mathbf{z}}$ be the projection of $\mathbf{z}$ on C, so that we have: $(\mathbf{z} - \hat{\mathbf{z}})^t(\mathbf{x} - \hat{\mathbf{z}})^t \leq 0$, for all $\mathbf{x} \in C$. Taking $\mathbf{x} = \mathbf{z}$ and $\mathbf{x} = 2\mathbf{z}$, we deduce: $(\mathbf{z} - \hat{\mathbf{z}})^t\hat{\mathbf{z}} = 0$, so that we have:

$$(\mathbf{z} - \hat{\mathbf{z}})^t\mathbf{x} \leq 0,$$

for all $\mathbf{x} \in C$.

□

Polar cone theorem

Theorem 12

For a subset $C \subseteq \mathbb{R}^d$, we have:

$$C^{0^0} = \overline{\text{ConicalHull}(C)}.$$

In particular, if C is closed and convex, then we have:

$$C^{0^0} = C.$$

Proof when C is closed and convex.

Since we have $\mathbf{y}^t\mathbf{x} \leq 0$ for all $\mathbf{y} \in C^0$ and all $\mathbf{x} \in C$ it follows that we have $C \subseteq C^{0^0}$. To prove the reverse inclusion, take $\mathbf{z} \in C^{0^0}$. Let $\hat{\mathbf{z}}$ be the projection of $\mathbf{z}$ on C, so that we have: $(\mathbf{z} - \hat{\mathbf{z}})^t(\mathbf{x} - \hat{\mathbf{z}})^t \leq 0$, for all $\mathbf{x} \in C$. Taking $\mathbf{x} = \mathbf{z}$ and $\mathbf{x} = 2\mathbf{z}$, we deduce: $(\mathbf{z} - \hat{\mathbf{z}})^t\hat{\mathbf{z}} = 0$, so that we have:

$$(\mathbf{z} - \hat{\mathbf{z}})^t\mathbf{x} \leq 0,$$

for all $\mathbf{x} \in C$. Therefore, we have $\mathbf{z} - \hat{\mathbf{z}} \in C^0$. Since $\mathbf{z} \in C^{0^0}$, we deduce: $(\mathbf{z} - \hat{\mathbf{z}})^t\mathbf{z} \leq 0$. Subtracting $(\mathbf{z} - \hat{\mathbf{z}})^t\hat{\mathbf{z}} = 0$ yields that $\|\mathbf{z} - \hat{\mathbf{z}}\|^2 = 0$, that is, $\mathbf{z} = \hat{\mathbf{z}}$, thus $\mathbf{z} \in C$, implying $C^{0^0} \subseteq C$. □

Plan

Polyhedra

Definition 13

1. A subset $P \subseteq \mathbb{R}^d$ is a **convex polyhedron** (or simply a polyhedron) if

$$P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$$

holds, for a matrix $A \in \mathbb{R}^{m \times d}$ and a vector $\mathbf{b} \in \mathbb{R}^m$, where $m \in \mathbb{N}_{>0}$.

Polyhedra

Definition 13

1. A subset $P \subseteq \mathbb{R}^d$ is a **convex polyhedron** (or simply a polyhedron) if

$$P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$$

holds, for a matrix $A \in \mathbb{R}^{m \times d}$ and a vector $\mathbf{b} \in \mathbb{R}^m$, where $m \in \mathbb{N}_{>0}$.

2. we call the linear system $\{A\mathbf{x} \leq \mathbf{b}\}$ an **H-representation** of P and denote by $\mathrm{Polyhedron}(A, \mathbf{b})$ the polyhedron P, that is, the solution set of the system of linear inequalities $A\mathbf{x} \leq \mathbf{b}$.

Polyhedra

Definition 13

1. A subset $P \subseteq \mathbb{R}^d$ is a **convex polyhedron** (or simply a polyhedron) if
$$P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$$
holds, for a matrix $A \in \mathbb{R}^{m \times d}$ and a vector $\mathbf{b} \in \mathbb{R}^m$, where $m \in \mathbb{N}_{>0}$.
2. we call the linear system $\{A\mathbf{x} \leq \mathbf{b}\}$ an **H-representation** of P and denote by $\mathrm{Polyhedron}(A, \mathbf{b})$ the polyhedron P, that is, the solution set of the system of linear inequalities $A\mathbf{x} \leq \mathbf{b}$.

Remark 1

1. *A polyhedron is the intersection of finitely many affine half-spaces.*

Polyhedra

Definition 13

1. A subset $P \subseteq \mathbb{R}^d$ is a **convex polyhedron** (or simply a polyhedron) if
$$P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$$
holds, for a matrix $A \in \mathbb{R}^{m \times d}$ and a vector $\mathbf{b} \in \mathbb{R}^m$, where $m \in \mathbb{N}_{>0}$.
2. we call the linear system $\{A\mathbf{x} \leq \mathbf{b}\}$ an **H-representation** of P and denote by $\mathrm{Polyhedron}(A, \mathbf{b})$ the polyhedron P, that is, the solution set of the system of linear inequalities $A\mathbf{x} \leq \mathbf{b}$.

Remark 1

1. *A polyhedron is the intersection of finitely many affine half-spaces.*
2. *Therefore, a polyhedron is both closed and convex.*

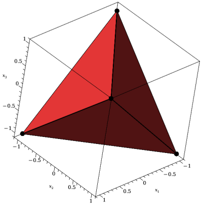
Polyhedra

Definition 13

1. A subset $P \subseteq \mathbb{R}^d$ is a **convex polyhedron** (or simply a polyhedron) if
$$P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$$
holds, for a matrix $A \in \mathbb{R}^{m \times d}$ and a vector $\mathbf{b} \in \mathbb{R}^m$, where $m \in \mathbb{N}_{>0}$.
2. we call the linear system $\{A\mathbf{x} \leq \mathbf{b}\}$ an **H-representation** of P and denote by Polyhedron$(A, \mathbf{b})$ the polyhedron P, that is, the solution set of the system of linear inequalities $A\mathbf{x} \leq \mathbf{b}$.

Remark 1

1. *A polyhedron is the intersection of finitely many affine half-spaces.*
2. *Therefore, a polyhedron is both closed and convex.*



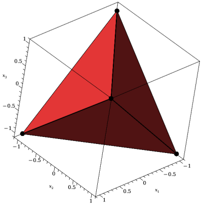
Polyhedra

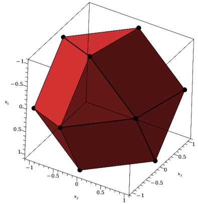
Definition 13

1. A subset $P \subseteq \mathbb{R}^d$ is a **convex polyhedron** (or simply a polyhedron) if
$$P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$$
holds, for a matrix $A \in \mathbb{R}^{m \times d}$ and a vector $\mathbf{b} \in \mathbb{R}^m$, where $m \in \mathbb{N}_{>0}$.
2. we call the linear system $\{A\mathbf{x} \leq \mathbf{b}\}$ an **H-representation** of P and denote by Polyhedron$(A, \mathbf{b})$ the polyhedron P, that is, the solution set of the system of linear inequalities $A\mathbf{x} \leq \mathbf{b}$.

Remark 1

1. *A polyhedron is the intersection of finitely many affine half-spaces.*
2. *Therefore, a polyhedron is both closed and convex.*





Implicit or redundant equalities (1/3)

Notations 4

1. *Let again* $P := \text{Polyhedron}(A, \mathbf{b})$.

Implicit or redundant equalities (1/3)

Notations 4

1. *Let again* $P := \text{Polyhedron}(A, \mathbf{b})$.
2. *Let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\beta \in \mathbb{R}$ *such that the inequality* $\mathbf{c}^t\mathbf{x} \leq \beta$ *is one of the inequalities of* $A\mathbf{x} \leq \mathbf{b}$.

Implicit or redundant equalities (1/3)

Notations 4

1. *Let again* $P := \text{Polyhedron}(A, \mathbf{b})$.
2. *Let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\beta \in \mathbb{R}$ *such that the inequality* $\mathbf{c}^t\mathbf{x} \le \beta$ *is one of the inequalities of* $A\mathbf{x} \le \mathbf{b}$.

Definition 14

1. We say that $\mathbf{c}^t\mathbf{x} \le \beta$ is an **redundant inequality** of $A\mathbf{x} \le \mathbf{b}$ if it is implied by the other inequalities defining P.

Implicit or redundant equalities (1/3)

Notations 4

1. *Let again* $P := \text{Polyhedron}(A, \mathbf{b})$.
2. *Let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\beta \in \mathbb{R}$ *such that the inequality* $\mathbf{c}^t\mathbf{x} \leq \beta$ *is one of the inequalities of* $A\mathbf{x} \leq \mathbf{b}$.

Definition 14

1. We say that $\mathbf{c}^t\mathbf{x} \leq \beta$ is an **redundant inequality** of $A\mathbf{x} \leq \mathbf{b}$ if it is implied by the other inequalities defining P.
2. We say that $\mathbf{c}^t\mathbf{x} \leq \beta$ is an **implicit equality** in $A\mathbf{x} \leq \mathbf{b}$ if for all $x \in \mathbb{R}^d$ we have

$$A\mathbf{x} \leq \mathbf{b} \implies \mathbf{c}^t\mathbf{x} = \beta.$$

Implicit or redundant equalities (1/3)

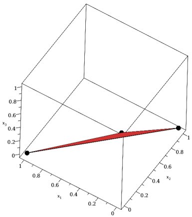
Notations 4

1. *Let again* $P := \text{Polyhedron}(A, \mathbf{b})$*.*
2. *Let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\beta \in \mathbb{R}$ *such that the inequality* $\mathbf{c}^t\mathbf{x} \leq \beta$ *is one of the inequalities of* $A\mathbf{x} \leq \mathbf{b}$*.*

Definition 14

1. We say that $\mathbf{c}^t\mathbf{x} \leq \beta$ is an **redundant inequality** of $A\mathbf{x} \leq \mathbf{b}$ if it is implied by the other inequalities defining P.
2. We say that $\mathbf{c}^t\mathbf{x} \leq \beta$ is an **implicit equality** in $A\mathbf{x} \leq \mathbf{b}$ if for all $x \in \mathbb{R}^d$ we have

$$A\mathbf{x} \leq \mathbf{b} \implies \mathbf{c}^t\mathbf{x} = \beta.$$



Implicit or redundant equalities (2/3)

Notations 5

Following [18], we denote by $A^{=}$ (resp. A^{+}) and $\mathbf{b}^{=}$ (resp. $\mathbf{b}^{+}$) the rows of A and $\mathbf{b}$ corresponding to the implicit (resp. non-implicit) equalities.

Implicit or redundant equalities (2/3)

Notations 5

Following [18], we denote by $A^=$ (resp. A^+) and $\mathbf{b}^=$ (resp. $\mathbf{b}^+$) the rows of A and $\mathbf{b}$ corresponding to the implicit (resp. non-implicit) equalities.

Proposition 6

If P is not empty, then there exists $\mathbf{x} \in P$ satisfying both

$$A^= \mathbf{x} = \mathbf{b}^= \quad \text{and} \quad A^+ \mathbf{x} < \mathbf{b}^+. \tag{2.1}$$

Implicit or redundant equalities (2/3)

Notations 5

Following [18], we denote by $A^{=}$ *(resp.* A^{+}*) and* $\mathbf{b}^{=}$ *(resp.* $\mathbf{b}^{+}$*) the rows of* A *and* $\mathbf{b}$ *corresponding to the implicit (resp. non-implicit) equalities.*

Proposition 6

If P *is not empty, then there exists* $\mathbf{x} \in P$ *satisfying both*

$$A^{=}\mathbf{x} \ = \ \mathbf{b}^{=} \ \text{ and } \ A^{+}\mathbf{x} \ < \ \mathbf{b}^{+}. \qquad (2.1)$$

Proof.

Assume P is not empty and has at least one non-implicit equality. Denote by $\mathbf{c}_1^t\mathbf{x} \leq \beta_1, \ldots, \mathbf{c}_e^t\mathbf{x} \leq \beta_e$. For each $1 \leq i \leq e$, there exists $\mathbf{x}_i \in P$ so that we have $\mathbf{c}_e^t\mathbf{x}_i \leq \beta_e$. Define:

$$\mathbf{x} \ = \ \frac{\mathbf{x}_1 + \cdots + \mathbf{x}_e}{e}.$$

Then $\mathbf{x}$ satisfies Equation 2.1 null. □

Implicit or redundant equalities (3/3)

Proposition 7

If Polyhedron$(A, \mathbf{b})$ *is not empty, then we have:*

$$\begin{array}{rcl} \text{AffineHull}(\text{Polyhedron}(A, \mathbf{b})) & = & \{\mathbf{x} \in \mathbb{R}^d \mid A^{=}\mathbf{x} = \mathbf{b}^{=}\} \\ & = & \{\mathbf{x} \in \mathbb{R}^d \mid A^{=}\mathbf{x} \leq \mathbf{b}^{=}\}. \end{array}$$

Implicit or redundant equalities (3/3)

Proposition 7

If Polyhedron$(A, \mathbf{b})$ *is not empty, then we have:*

$$\begin{array}{rcl} \mathrm{AffineHull}(\mathrm{Polyhedron}(A, \mathbf{b})) & = & \{\mathbf{x} \in \mathbb{R}^d \mid A^{=}\mathbf{x} = \mathbf{b}^{=}\} \\ & = & \{\mathbf{x} \in \mathbb{R}^d \mid A^{=}\mathbf{x} \leq \mathbf{b}^{=}\}. \end{array}$$

Proof.

Let $\mathbf{x}_1, \ldots, \mathbf{x}_e \in$ Polyhedron$(A, \mathbf{b})$, let $\lambda_1, \ldots, \lambda_e \in \mathbb{R}$, and let $\mathbf{c}^t\mathbf{x} \leq \beta$ be an implicit equality. Since $\mathbf{c}^t\mathbf{x}_i = \beta$ holds for all $1 \leq i \leq e$, we have:

$$\mathbf{c}^t \left(\Sigma_{i=1}^{e} \lambda_i \mathbf{x}_i\right) = \beta.$$

The inclusion $\subseteq$ follows.

□

Implicit or redundant equalities (3/3)

Proposition 7

If Polyhedron$(A, \mathbf{b})$ *is not empty, then we have:*

$$\text{AffineHull}(\text{Polyhedron}(A, \mathbf{b})) = \{\mathbf{x} \in \mathbb{R}^d \mid A^= \mathbf{x} = \mathbf{b}^=\} = \{\mathbf{x} \in \mathbb{R}^d \mid A^= \mathbf{x} \le \mathbf{b}^=\}.$$

Proof.

Let $\mathbf{x}_1, \ldots, \mathbf{x}_e \in$ Polyhedron$(A, \mathbf{b})$, let $\lambda_1, \ldots, \lambda_e \in \mathbb{R}$, and let $\mathbf{c}^t \mathbf{x} \le \beta$ be an implicit equality. Since $\mathbf{c}^t \mathbf{x}_i = \beta$ holds for all $1 \le i \le e$, we have:

$$\mathbf{c}^t \left(\Sigma_{i=1}^{e} \lambda_i \mathbf{x}_i \right) = \beta.$$

The inclusion $\subseteq$ follows. Conversely, let $\mathbf{x}_0$ satisfy $A^= \mathbf{x} \le \mathbf{b}^=$. Let $\mathbf{x}_1 \in$ Polyhedron$(A, \mathbf{b})$. Using Proposition 6 null, we can assume that $A^+ \mathbf{x}_1 < \mathbf{b}^+$ holds. If $\mathbf{x}_0 = \mathbf{x}_1$, then we have $\mathbf{x}_0 \in$ Polyhedron$(A, \mathbf{b})$ and thus $\mathbf{x}_0 \in$ AffineHull(Polyhedron$(A, \mathbf{b})$). Otherwise, using a continuity argument, the segment joining $\mathbf{x}_0$ and $\mathbf{x}_1$ contains more than one point in Polyhedron$(A, \mathbf{b})$, and thus more than one point in AffineHull(Polyhedron$(A, \mathbf{b})$). Therefore, the whole segment $[\mathbf{x}_0, \mathbf{x}_1]$ is necessarily contained in that latter set, which proves the inclusion $\supseteq$. □

The faces of a polyhedral set (1/2)

Notations 6

Let again $P := \text{Polyhedron}(A, \mathbf{b})$ *be a polyhedron of* $\mathbb{R}^d$.

Proposition 8

A non-empty subset $F \subseteq P$ *is a* **face** *of* P *if* $F = \{\mathbf{x} \in P | A'\mathbf{x} = \mathbf{b}'\}$ *for some subsystem* $A'\mathbf{x} \leq \mathbf{b}'$ *of* $A\mathbf{x} \leq \mathbf{b}$.

Definition 15

1. A face of P, distinct from P, and with maximum dimension is a **facet** of P.
2. The **lineality space** of P is $\{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} = \vec{0}\}$ and P is said **pointed** if its lineality space has dimension zero.
3. For a pointed polyhedron P, the inclusion-minimal faces are the **vertices** of P, that is, its 0-dimensional faces.

The faces of a polyhedral set (2/2)

Theorem 16

If Polyhedron$(A, \mathbf{b})$ *is full-dimensional (that is, has dimension d) and the the system $A\mathbf{x} \leq \mathbf{b}$ has no redundant inequalities, then the inequalities of that system are in 1-to-1 correspondence with the facets of P.*

The faces of a polyhedral set (2/2)

Theorem 16

If Polyhedron$(A, \mathbf{b})$ is full-dimensional (that is, has dimension d) and the the system $A\mathbf{x} \leq \mathbf{b}$ has no redundant inequalities, then the inequalities of that system are in 1-to-1 correspondence with the facets of P.

Theorem 17

1. *The faces of P, when ordered by the set theoretic inclusion, form a lattice L which enjoys three important properties (that are not true in general for an arbitrary lattice):*

skip slide

The faces of a polyhedral set (2/2)

Theorem 16

If Polyhedron$(A, \mathbf{b})$ is full-dimensional (that is, has dimension d) and the the system $A\mathbf{x} \leq \mathbf{b}$ has no redundant inequalities, then the inequalities of that system are in 1-to-1 correspondence with the facets of P.

Theorem 17

1. *The faces of P, when ordered by the set theoretic inclusion, form a lattice L which enjoys three important properties (that are not true in general for an arbitrary lattice):*
   a. *L is graded (that is, it admits a rank function),*

skip slide

The faces of a polyhedral set (2/2)

Theorem 16

If Polyhedron$(A, \mathbf{b})$ *is full-dimensional (that is, has dimension d) and the the system $A\mathbf{x} \leq \mathbf{b}$ has no redundant inequalities, then the inequalities of that system are in 1-to-1 correspondence with the facets of P.*

Theorem 17

1. *The faces of P, when ordered by the set theoretic inclusion, form a lattice L which enjoys three important properties (that are not true in general for an arbitrary lattice):*
   - a. *L is graded (that is, it admits a rank function),*
   - b. *L is ranked (that is, its maximal chains have the same cardinality),*

The faces of a polyhedral set (2/2)

Theorem 16

If Polyhedron$(A, \mathbf{b})$ is full-dimensional (that is, has dimension d) and the the system $A\mathbf{x} \leq \mathbf{b}$ has no redundant inequalities, then the inequalities of that system are in 1-to-1 correspondence with the facets of P.

Theorem 17

1. *The faces of P, when ordered by the set theoretic inclusion, form a lattice L which enjoys three important properties (that are not true in general for an arbitrary lattice):*
   a. *L is graded (that is, it admits a rank function),*
   b. *L is ranked (that is, its maximal chains have the same cardinality),*
   c. *if the ranks of two faces $a > b$ differ by 2, then there are exactly 2 faces that lie strictly between a and b.*

skip slide

The faces of a polyhedral set (2/2)

Theorem 16

If Polyhedron$(A, \mathbf{b})$ is full-dimensional (that is, has dimension d) and the the system $A\mathbf{x} \leq \mathbf{b}$ has no redundant inequalities, then the inequalities of that system are in 1-to-1 correspondence with the facets of P.

Theorem 17

1. *The faces of P, when ordered by the set theoretic inclusion, form a lattice L which enjoys three important properties (that are not true in general for an arbitrary lattice):*
   - a. *L is graded (that is, it admits a rank function),*
   - b. *L is ranked (that is, its maximal chains have the same cardinality),*
   - c. *if the ranks of two faces $a > b$ differ by 2, then there are exactly 2 faces that lie strictly between a and b.*
2. *the face lattice of a polytope can be uniquely determined from its facets, its vertices and its vertex-facet incidences.*

skip slide

Linear programming (review)

Notations 7

1. *Let* $A \in \mathbb{R}^{m \times d}$ *be a matrix, let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\mathbf{b} \in \mathbb{R}^m$ *be two vectors*

Linear programming (review)

Notations 7

1. *Let* $A \in \mathbb{R}^{m \times d}$ *be a matrix, let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\mathbf{b} \in \mathbb{R}^m$ *be two vectors*
2. *Consider the linear program*

$$\text{Minimize} \quad \mathbf{c}^t\mathbf{x} \quad \text{subject} \quad \text{to} \quad A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}.$$

Linear programming (review)

Notations 7

1. *Let* $A \in \mathbb{R}^{m \times d}$ *be a matrix, let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\mathbf{b} \in \mathbb{R}^m$ *be two vectors*
2. *Consider the linear program*

$$\text{Minimize} \quad \mathbf{c}^t\mathbf{x} \quad \text{subject} \quad \text{to} \quad A\mathbf{x} \le \mathbf{b}, \mathbf{x} \ge \mathbf{0}.$$

3. *and its dual:*

$$\text{Maximize} \quad \mathbf{b}^t\mathbf{y} \quad \text{subject} \quad \text{to} \quad A^t\mathbf{y} \le \mathbf{c}, \mathbf{y} \ge \mathbf{0}.$$

Linear programming (review)

Notations 7

1. *Let* $A \in \mathbb{R}^{m \times d}$ *be a matrix, let* $\mathbf{c} \in \mathbb{R}^d$ *and* $\mathbf{b} \in \mathbb{R}^m$ *be two vectors*
2. *Consider the linear program*

$$\text{Minimize} \quad \mathbf{c}^t\mathbf{x} \quad \text{subject} \quad \text{to} \quad A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}.$$

3. *and its dual:*

$$\text{Maximize} \quad \mathbf{b}^t\mathbf{y} \quad \text{subject} \quad \text{to} \quad A^t\mathbf{y} \leq \mathbf{c}, \mathbf{y} \geq \mathbf{0}.$$

Proposition 9

1. *Weak duality: If both programs have feasible solutions, then*

$$\max_{\mathbf{y}} \mathbf{b}^t\mathbf{y} \;\leq\; \min_{\mathbf{x}} \mathbf{c}^t\mathbf{x}.$$

Linear programming (review)

Notations 7

1. *Let* $A \in \mathbb{R}^{m\times d}$ *be a matrix, let* $\mathbf{c} \in \mathbb{R}^{d}$ *and* $\mathbf{b} \in \mathbb{R}^{m}$ *be two vectors*
2. *Consider the linear program*

$$\text{Minimize } \mathbf{c}^t\mathbf{x} \text{ subject to } A\mathbf{x} \le \mathbf{b}, \mathbf{x} \ge \mathbf{0}.$$

3. *and its dual:*

$$\text{Maximize } \mathbf{b}^t\mathbf{y} \text{ subject to } A^t\mathbf{y} \le \mathbf{c}, \mathbf{y} \ge \mathbf{0}.$$

Proposition 9

1. *Weak duality: If both programs have feasible solutions, then*

$$\max_{\mathbf{y}}\mathbf{b}^t\mathbf{y} \;\le\; \min_{\mathbf{x}}\mathbf{c}^t\mathbf{x}.$$

2. *Strong duality: if one of the two problems has an optimal solution, so does the other one and the bounds given by the weak duality theorem are tight.*

Linear programming (review)

Notations 7

1. *Let $A \in \mathbb{R}^{m \times d}$ be a matrix, let $\mathbf{c} \in \mathbb{R}^d$ and $\mathbf{b} \in \mathbb{R}^m$ be two vectors*
2. *Consider the linear program*

$$\text{Minimize } \ \mathbf{c}^t\mathbf{x} \ \text{ subject to } \ A\mathbf{x} \le \mathbf{b}, \mathbf{x} \ge \mathbf{0}.$$

3. *and its dual:*

$$\text{Maximize } \ \mathbf{b}^t\mathbf{y} \ \text{ subject to } \ A^t\mathbf{y} \le \mathbf{c}, \mathbf{y} \ge \mathbf{0}.$$

Proposition 9

1. *Weak duality: If both programs have feasible solutions, then*

$$\max_{\mathbf{y}} \mathbf{b}^t\mathbf{y} \ \le \ \min_{\mathbf{x}} \mathbf{c}^t\mathbf{x}.$$

2. *Strong duality: if one of the two problems has an optimal solution, so does the other one and the bounds given by the weak duality theorem are tight.*

Remark 2

The same results hold if the constraints are $A\mathbf{x} = \mathbf{b}, \mathbf{x} \ge \mathbf{0}$ and $A^t\mathbf{y} \le \mathbf{c}$, respectively.

Fundamental theorem of linear inequalities

Theorem 18

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* r *the rank of the* $d \times (m+1)$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}$*. Then exactly one of the following statements holds:*

Fundamental theorem of linear inequalities

Theorem 18

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* r *the rank of the* $d \times (m+1)$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}$*. Then exactly one of the following statements holds:*

1. *There exists* s *linearly independent vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *and* s *numbers* $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ *so that* $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$*,*

Fundamental theorem of linear inequalities

Theorem 18

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* r *the rank of the* $d \times (m+1)$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}$*. Then exactly one of the following statements holds:*

1. *There exists* s *linearly independent vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *and* s *numbers* $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ *so that* $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$*,*
2. *There exists a hyperplane* $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{c}^t \mathbf{x} = 0\}$ *containing* $r-1$ *linearly independent vectors from* $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *such that we have:*

Fundamental theorem of linear inequalities

Theorem 18

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* r *the rank of the* $d \times (m+1)$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}$*. Then exactly one of the following statements holds:*

1. *There exists* s *linearly independent vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *and* s *numbers* $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ *so that* $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$*,*
2. *There exists a hyperplane* $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{c}^t \mathbf{x} = 0\}$ *containing* $r-1$ *linearly independent vectors from* $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *such that we have:*
    - a. $\mathbf{c}^t \mathbf{b} < 0$,

Fundamental theorem of linear inequalities

Theorem 18

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$, Denote by r the rank of the $d \times (m+1)$ matrix whose columns are $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}$. Then exactly one of the following statements holds:

1. *There exists s linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ and s numbers $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ so that $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$,*
2. *There exists a hyperplane $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{c}^t \mathbf{x} = 0\}$ containing $r-1$ linearly independent vectors from $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ such that we have:*
    - a. $\mathbf{c}^t \mathbf{b} < 0$,
    - b. $\mathbf{c}^t \mathbf{a}_i \geq 0$, *for all $1 \leq i \leq m$.*

Fundamental theorem of linear inequalities

Theorem 18

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* r *the rank of the* $d \times (m+1)$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}$*. Then exactly one of the following statements holds:*

1. *There exists* s *linearly independent vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *and* s *numbers* $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ *so that* $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$*,*
2. *There exists a hyperplane* $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{c}^t \mathbf{x} = 0\}$ *containing* $r - 1$ *linearly independent vectors from* $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *such that we have:*
   a. $\mathbf{c}^t \mathbf{b} < 0$,
   b. $\mathbf{c}^t \mathbf{a}_i \geq 0$*, for all* $1 \leq i \leq m$*.*

   Moreover, if $\mathbf{a}_1, \ldots, \mathbf{a}_m$ *are rational then* $\mathbf{c}$ *is rational as well.*

Fundamental theorem of linear inequalities

Theorem 18

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* r *the rank of the* $d \times (m+1)$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}$*. Then exactly one of the following statements holds:*

1. *There exists* s *linearly independent vectors* $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *and* s *numbers* $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ *so that* $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$*,*
2. *There exists a hyperplane* $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{c}^t \mathbf{x} = 0\}$ *containing* $r-1$ *linearly independent vectors from* $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ *such that we have:*
   - a. $\mathbf{c}^t \mathbf{b} < 0$,
   - b. $\mathbf{c}^t \mathbf{a}_i \geq 0$*, for all* $1 \leq i \leq m$*.*

   Moreover, if $\mathbf{a}_1, \ldots, \mathbf{a}_m$ *are rational then* $\mathbf{c}$ *is rational as well.*

Proof.

1. A constructive proof based on the simplex algorithm in Schrijver's book [18].
2. For a shorter proof, one can weaken the above statement and proving Gordan's theorem, instead.
3. The proof of Gordan's theorem is very similar to that of the bipolar theorem.

Farkas lemma

Theorem 19

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* A *the* $m \times d$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m$*. Then exactly one of the following statements holds:*

Farkas lemma

Theorem 19

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* A *the* $m \times d$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m$*. Then exactly one of the following statements holds:*

1. *the system* $A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ *has solutions,*

Farkas lemma

Theorem 19

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* A *the* $m \times d$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m$*. Then exactly one of the following statements holds:*

1. *the system* $A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ *has solutions,*
2. *there exists* $\mathbf{y} \in \mathbb{R}^m$ *such that* $\mathbf{y}^t A \geq 0, \mathbf{y}^t b < 0.$

Farkas lemma

Theorem 19

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* A *the* $m \times d$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m$*. Then exactly one of the following statements holds:*

1. *the system* $A\mathbf{x} = \mathbf{b}, \ \mathbf{x} \geq \mathbf{0}$ *has solutions,*
2. *there exists* $\mathbf{y} \in \mathbb{R}^m$ *such that* $\mathbf{y}^t A \geq 0, \mathbf{y}^t b < 0$.

Proof.

We apply Theorem 18 null.

□

Farkas lemma

Theorem 19

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* A *the* $m \times d$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m$*. Then exactly one of the following statements holds:*

1. *the system* $A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ *has solutions,*
2. *there exists* $\mathbf{y} \in \mathbb{R}^m$ *such that* $\mathbf{y}^t A \geq 0, \mathbf{y}^t b < 0$.

Proof.

We apply Theorem 18 null. Suppose that here exists s linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ and s numbers $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ so that $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$. This is equivalent to say that the system $A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ has solutions.

□

Farkas lemma

Theorem 19

Let $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b} \in \mathbb{R}^d$*, Denote by* A *the* $m \times d$ *matrix whose columns are* $\mathbf{a}_1, \ldots, \mathbf{a}_m$*. Then exactly one of the following statements holds:*

1. *the system* $A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ *has solutions,*
2. *there exists* $\mathbf{y} \in \mathbb{R}^m$ *such that* $\mathbf{y}^t A \geq 0, \mathbf{y}^t b < 0$.

Proof.

We apply Theorem 18 null. Suppose that here exists s linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ and s numbers $\lambda_1, \ldots, \lambda_s \in \mathbb{R}_{\geq 0}$ so that $b = \lambda_1 \mathbf{v}_1 + \cdots + \lambda_s \mathbf{v}_s$. This is equivalent to say that the system $A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}$ has solutions.
Suppose now that there exists a hyperplane $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{c}^t\mathbf{x} = 0\}$ containing $r - 1$ linearly independent vectors from $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ such that we have $\mathbf{c}^t\mathbf{b} < 0$ and $\mathbf{c}^t\mathbf{a}_i \geq 0$, for all $1 \leq i \leq m$. Set $\mathbf{y} = \mathbf{c}$. Then, this implies that both $\mathbf{y}^t A \geq 0$ and $\mathbf{y}^t b < 0$ hold. □

Plan

Farkas–Minkowsi–Weyl theorem (1/4)

Theorem 20

A convex cone $C \subseteq \mathbb{R}^d$ is polyhedral if and only if it is finitely generated.

Farkas–Minkowsi–Weyl theorem (1/4)

Theorem 20

A convex cone $C \subseteq \mathbb{R}^d$ is polyhedral if and only if it is finitely generated.

Proof.

Assume there exists $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.

□

Farkas–Minkowsi–Weyl theorem (1/4)

Theorem 20

A convex cone $C \subseteq \mathbb{R}^d$ is polyhedral if and only if it is finitely generated.

Proof.

Assume there exists $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.

1. W.l.o.g. assume that $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \mathbb{R}^d$, otherwise do the proof in $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Thus we have $d \leq m$.

□

Farkas–Minkowsi–Weyl theorem (1/4)

Theorem 20

A convex cone $C \subseteq \mathbb{R}^d$ is polyhedral if and only if it is finitely generated.

Proof.

Assume there exists $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.

1. W.l.o.g. assume that $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \mathbb{R}^d$, otherwise do the proof in $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Thus we have $d \leq m$.
2. By the fundamental theorem of linear inequalities, the following statements are equivalent, for all $\mathbf{y} \in \mathbb{R}^d$:
   - a. $\mathbf{y} \in \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$,
   - b. for every hyperplane $\{\mathbf{x} \mid \mathbf{c}^t\mathbf{x} = 0\}$ containing $d-1$ linearly independent vectors from $\mathbf{x}_1, \ldots, \mathbf{x}_m$ so that $\mathbf{c}^t\mathbf{x}_{\geq}0$ holds for all $1 \leq i \leq m$, we have $\mathbf{c}^t\mathbf{y} \geq 0$.

□

Farkas–Minkowsi–Weyl theorem (1/4)

Theorem 20

A convex cone $C \subseteq \mathbb{R}^d$ is polyhedral if and only if it is finitely generated.

Proof.

Assume there exists $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.

1. W.l.o.g. assume that $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \mathbb{R}^d$, otherwise do the proof in $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Thus we have $d \leq m$.
2. By the fundamental theorem of linear inequalities, the following statements are equivalent, for all $\mathbf{y} \in \mathbb{R}^d$:
   - a. $\mathbf{y} \in \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$,
   - b. for every hyperplane $\{\mathbf{x} \mid \mathbf{c}^t\mathbf{x} = 0\}$ containing $d - 1$ linearly independent vectors from $\mathbf{x}_1, \ldots, \mathbf{x}_m$ so that $\mathbf{c}^t\mathbf{x}_{\geq}0$ holds for all $1 \leq i \leq m$, we have $\mathbf{c}^t\mathbf{y} \geq 0$.
3. Consider all hyperplanes spanned by $d - 1$ linearly independent vectors from $\mathbf{x}_1, \ldots, \mathbf{x}_m$ so that $\mathbf{c}^t\mathbf{x}_i \geq 0$ holds for all $1 \leq i \leq m$.

□

Farkas–Minkowsi–Weyl theorem (1/4)

Theorem 20

A convex cone $C \subseteq \mathbb{R}^d$ is polyhedral if and only if it is finitely generated.

Proof.

Assume there exists $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.

1. W.l.o.g. assume that $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \mathbb{R}^d$, otherwise do the proof in $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Thus we have $d \leq m$.
2. By the fundamental theorem of linear inequalities, the following statements are equivalent, for all $\mathbf{y} \in \mathbb{R}^d$:
   a. $\mathbf{y} \in \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$,
   b. for every hyperplane $\{\mathbf{x} \mid \mathbf{c}^t\mathbf{x} = 0\}$ containing $d-1$ linearly independent vectors from $\mathbf{x}_1, \ldots, \mathbf{x}_m$ so that $\mathbf{c}^t\mathbf{x}_{\geq}0$ holds for all $1 \leq i \leq m$, we have $\mathbf{c}^t\mathbf{y} \geq 0$.
3. Consider all hyperplanes spanned by $d-1$ linearly independent vectors from $\mathbf{x}_1, \ldots, \mathbf{x}_m$ so that $\mathbf{c}^t\mathbf{x}_i \geq 0$ holds for all $1 \leq i \leq m$.
4. The number of such hyperplanes is at most $N := \binom{m}{d-1}$.

□

Farkas–Minkowsi–Weyl theorem (1/4)

Theorem 20

A convex cone $C \subseteq \mathbb{R}^d$ is polyhedral if and only if it is finitely generated.

Proof.

Assume there exists $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.

1. W.l.o.g. assume that $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \mathbb{R}^d$, otherwise do the proof in $\text{Span}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Thus we have $d \leq m$.
2. By the fundamental theorem of linear inequalities, the following statements are equivalent, for all $\mathbf{y} \in \mathbb{R}^d$:
   - a. $\mathbf{y} \in \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$,
   - b. for every hyperplane $\{\mathbf{x} \mid \mathbf{c}^t\mathbf{x} = 0\}$ containing $d - 1$ linearly independent vectors from $\mathbf{x}_1, \ldots, \mathbf{x}_m$ so that $\mathbf{c}^t\mathbf{x}_{\geq}0$ holds for all $1 \leq i \leq m$, we have $\mathbf{c}^t\mathbf{y} \geq 0$.
3. Consider all hyperplanes spanned by $d - 1$ linearly independent vectors from $\mathbf{x}_1, \ldots, \mathbf{x}_m$ so that $\mathbf{c}^t\mathbf{x}_i \geq 0$ holds for all $1 \leq i \leq m$.
4. The number of such hyperplanes is at most $N := \binom{m}{d-1}$.
5. Let those hyperplanes be defined by $\mathbf{c}_1, \ldots, \mathbf{c}_N$.

□

Farkas–Minkowsi–Weyl theorem (2/4)

Proof.

We shall prove that we have:

$$\text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m) \; = \; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{x}\mathbf{c}_i^t \geq 0, 1 \leq i \leq N\}, \qquad (2.2)$$

which will imply that the cone C is polyhedral. We prove the two inclusions $\subseteq$ and $\supseteq$.

□

Farkas–Minkowsi–Weyl theorem (2/4)

Proof.

We shall prove that we have:

$$\text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m) \; = \; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{x}\mathbf{c}_i^t \geq 0, 1 \leq i \leq N\}, \qquad (2.2)$$

which will imply that the cone C is polyhedral. We prove the two inclusions $\subseteq$ and $\supseteq$.

□

Farkas–Minkowsi–Weyl theorem (2/4)

Proof.

We shall prove that we have:

$$\text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m) \;=\; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{x}\mathbf{c}_i^t \geq 0, 1 \leq i \leq N\}, \qquad (2.2)$$

which will imply that the cone C is polyhedral. We prove the two inclusions $\subseteq$ and $\supseteq$.

- $\subseteq$: This inclusion follows immediately from our above observation derived from the fundamental theorem of linear inequalities.

□

Farkas–Minkowsi–Weyl theorem (2/4)

Proof.

We shall prove that we have:

$$\text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{x}\mathbf{c}_i^t \geq 0, 1 \leq i \leq N\}, \qquad (2.2)$$

which will imply that the cone C is polyhedral. We prove the two inclusions $\subseteq$ and $\supseteq$.

- $\subseteq$: This inclusion follows immediately from our above observation derived from the fundamental theorem of linear inequalities.
- $\supseteq$: Consider $\mathbf{y} \in \mathbb{R}^d$ so that $\mathbf{y} \notin \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. From the same observation, there exists an hyperplane H_i among those defined by $\mathbf{c}_1, \ldots, \mathbf{c}_N$ such that $\mathbf{x}\mathbf{c}_i^t \geq 0$ does **not** hold.

□

Farkas–Minkowsi–Weyl theorem (2/4)

Proof.

We shall prove that we have:

$$\text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m) \; = \; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{x}\mathbf{c}_i^t \geq 0, 1 \leq i \leq N\}, \tag{2.2}$$

which will imply that the cone C is polyhedral. We prove the two inclusions $\subseteq$ and $\supseteq$.

- $\subseteq$: This inclusion follows immediately from our above observation derived from the fundamental theorem of linear inequalities.
- $\supseteq$: Consider $\mathbf{y} \in \mathbb{R}^d$ so that $\mathbf{y} \notin \text{ConicalHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. From the same observation, there exists an hyperplane H_i among those defined by $\mathbf{c}_1, \ldots, \mathbf{c}_N$ such that $\mathbf{x}\mathbf{c}_i^t \geq 0$ does **not** hold.

Assume now that exist $\mathbf{a}_1, \ldots, \mathbf{a}_m \in \mathbb{R}^d$ so that we have

$$C = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t\mathbf{x} \leq \mathbf{0}, \; 1 \leq i \leq m\}, \tag{2.3}$$

that is, assume C is polyhedral. □

Farkas–Minkowsi–Weyl theorem (3/4)

Proof.

1. From the first part of the proof, there exist vectors $\mathbf{b}_1, \ldots, \mathbf{b}_N \in \mathbb{R}^d$ such that we have:

$$\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq N\}, \quad (2.4)$$

□

Farkas–Minkowsi–Weyl theorem (3/4)

Proof.

1. From the first part of the proof, there exist vectors $\mathbf{b}_1, \ldots, \mathbf{b}_N \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) \; = \; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \leq \mathbf{0}, \; 1 \leq i \leq N\}, \quad (2.4)$$
2. Note that for all $1 \leq j \leq m$ and all $1 \leq i \leq N$, we have:
$$\mathbf{b}_i^t \mathbf{a}_i \leq 0, \quad (2.5)$$
since $\mathbf{a}_j, \in$ ConicalHull$(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ holds.

□

Farkas–Minkowsi–Weyl theorem (3/4)

Proof.

1. From the first part of the proof, there exist vectors $\mathbf{b}_1, \ldots, \mathbf{b}_N \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) \;=\; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq N\}, \quad (2.4)$$
2. Note that for all $1 \leq j \leq m$ and all $1 \leq i \leq N$, we have:
$$\mathbf{b}_i^t \mathbf{a}_i \leq 0, \quad (2.5)$$
since $\mathbf{a}_j, \in$ ConicalHull$(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ holds.
3. We will show that C is a finitely generated, by proving:
$$C \;=\; \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N). \quad (2.6)$$

□

Farkas–Minkowsi–Weyl theorem (3/4)

Proof.

1. From the first part of the proof, there exist vectors $\mathbf{b}_1, \ldots, \mathbf{b}_N \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) \ = \ \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \leq \mathbf{0}, \ 1 \leq i \leq N\}, \qquad (2.4)$$
2. Note that for all $1 \leq j \leq m$ and all $1 \leq i \leq N$, we have:
$$\mathbf{b}_i^t \mathbf{a}_i \leq 0, \qquad (2.5)$$
since $\mathbf{a}_j, \in \text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ holds.
3. We will show that C is a finitely generated, by proving:
$$C \ = \ \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N). \qquad (2.6)$$
4. We prove $\text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N) \ \subseteq \ \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0}, \ 1 \leq i \leq m\}$.

□

Farkas–Minkowsi–Weyl theorem (3/4)

Proof.

1. From the first part of the proof, there exist vectors $\mathbf{b}_1, \ldots, \mathbf{b}_N \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) \;=\; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \le \mathbf{0},\ 1 \le i \le N\}, \quad (2.4)$$
2. Note that for all $1 \le j \le m$ and all $1 \le i \le N$, we have:
$$\mathbf{b}_i^t \mathbf{a}_i \le 0, \quad (2.5)$$
since $\mathbf{a}_j, \in$ ConicalHull$(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ holds.
3. We will show that C is a finitely generated, by proving:
$$C \;=\; \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N). \quad (2.6)$$
4. We prove ConicalHull$(\mathbf{b}_1, \ldots, \mathbf{b}_N) \;\subseteq\; \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \le \mathbf{0},\ 1 \le i \le m\}$.
5. Observe that $\mathbf{b}_i \in C$, since $\mathbf{b}_i^t \mathbf{a}_i \le 0$ holds for all $1 \le j \le m$ and all $1 \le i \le N$.

□

Farkas–Minkowsi–Weyl theorem (3/4)

Proof.

1. From the first part of the proof, there exist vectors $\mathbf{b}_1, \ldots, \mathbf{b}_N \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq N\}, \quad (2.4)$$
2. Note that for all $1 \leq j \leq m$ and all $1 \leq i \leq N$, we have:
$$\mathbf{b}_i^t \mathbf{a}_i \leq 0, \quad (2.5)$$
since $\mathbf{a}_j, \in$ ConicalHull$(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ holds.
3. We will show that C is a finitely generated, by proving:
$$C = \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N). \quad (2.6)$$
4. We prove ConicalHull$(\mathbf{b}_1, \ldots, \mathbf{b}_N) \subseteq \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\}$.
5. Observe that $\mathbf{b}_i \in C$, since $\mathbf{b}_i^t \mathbf{a}_i \leq 0$ holds for all $1 \leq j \leq m$ and all $1 \leq i \leq N$.
6. It follows that all linear combination of $\mathbf{b}_1, \ldots, \mathbf{b}_N$ with non-negative coefficients are in C. This proves the inclusion.

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \subseteq \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \subseteq \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

1. So let $\mathbf{y} \in C$ and $\mathbf{y} \notin \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$.

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \subseteq$ ConicalHull$(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

1. So let $\mathbf{y} \in C$ and $\mathbf{y} \notin$ ConicalHull$(\mathbf{b}_1, \ldots, \mathbf{b}_N)$.
2. From the first part of the proof, there exist vectors $\mathbf{w}_1, \ldots, \mathbf{w}_r \in \mathbb{R}^d$ such that we have:

$$\text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{w}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq r\}, \quad (2.7)$$

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \subseteq \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

1. So let $\mathbf{y} \in C$ and $\mathbf{y} \notin \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$.
2. From the first part of the proof, there exist vectors $\mathbf{w}_1, \ldots, \mathbf{w}_r \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{w}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq r\}, \quad (2.7)$$
3. Consequently, there exists some $1 \leq i \leq r$, such that $\mathbf{w}_i^t \mathbf{y} > 0$.

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t\mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \ \subseteq\ \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

1. So let $\mathbf{y} \in C$ and $\mathbf{y} \notin \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$.
2. From the first part of the proof, there exist vectors $\mathbf{w}_1, \ldots, \mathbf{w}_r \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N) \ = \ \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{w}_i^t\mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq r\}, \qquad (2.7)$$
3. Consequently, there exists some $1 \leq i \leq r$, such that $\mathbf{w}_i^t\mathbf{y} > 0$.
4. By definition of $\mathbf{w}_1, \ldots, \mathbf{w}_r$ we have: $\mathbf{w}_i^t\mathbf{b}_j \leq 0$, for all $1 \leq j \leq N$.

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t\mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \subseteq \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

1. So let $\mathbf{y} \in C$ and $\mathbf{y} \notin \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$.
2. From the first part of the proof, there exist vectors $\mathbf{w}_1, \ldots, \mathbf{w}_r \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{w}_i^t\mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq r\}, \quad (2.7)$$
3. Consequently, there exists some $1 \leq i \leq r$, such that $\mathbf{w}_i^t\mathbf{y} > 0$.
4. By definition of $\mathbf{w}_1, \ldots, \mathbf{w}_r$ we have: $\mathbf{w}_i^t\mathbf{b}_j \leq 0$, for all $1 \leq j \leq N$.
5. Since $\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t\mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq N\}$, there exist $\lambda_1, \ldots, \lambda_m \in \mathbb{R}_{\geq 0}$ such that we have: $\mathbf{w}_i^t = \Sigma_{k=1}^m \lambda_k \mathbf{a}_k$.

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \subseteq \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

1. So let $\mathbf{y} \in C$ and $\mathbf{y} \notin \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$.
2. From the first part of the proof, there exist vectors $\mathbf{w}_1, \ldots, \mathbf{w}_r \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{w}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq r\}, \quad (2.7)$$
3. Consequently, there exists some $1 \leq i \leq r$, such that $\mathbf{w}_i^t \mathbf{y} > 0$.
4. By definition of $\mathbf{w}_1, \ldots, \mathbf{w}_r$ we have: $\mathbf{w}_i^t \mathbf{b}_j \leq 0$, for all $1 \leq j \leq N$.
5. Since $\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq N\}$, there exist $\lambda_1, \ldots, \lambda_m \in \mathbb{R}_{\geq 0}$ such that we have: $\mathbf{w}_i^t = \sum_{k=1}^m \lambda_k \mathbf{a}_k$.
6. Since $\mathbf{y} \in C$ and $C = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\}$, we deduce
$$\mathbf{w}_i^t \mathbf{y} = \sum_{k=1}^m \lambda_k \mathbf{a}_k^t \mathbf{w}_i \leq 0. \quad (2.8)$$

□

Farkas–Minkowsi–Weyl theorem (4/4)

Proof.

We prove $\{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\} \subseteq \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$, by contradiction.

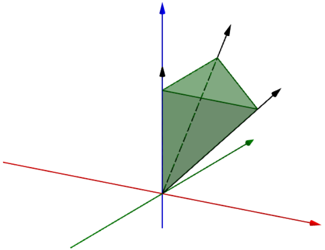
1. So let $\mathbf{y} \in C$ and $\mathbf{y} \notin \text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N)$.
2. From the first part of the proof, there exist vectors $\mathbf{w}_1, \ldots, \mathbf{w}_r \in \mathbb{R}^d$ such that we have:
$$\text{ConicalHull}(\mathbf{b}_1, \ldots, \mathbf{b}_N) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{w}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq r\}, \quad (2.7)$$
3. Consequently, there exists some $1 \leq i \leq r$, such that $\mathbf{w}_i^t \mathbf{y} > 0$.
4. By definition of $\mathbf{w}_1, \ldots, \mathbf{w}_r$ we have: $\mathbf{w}_i^t \mathbf{b}_j \leq 0$, for all $1 \leq j \leq N$.
5. Since $\text{ConicalHull}(\mathbf{a}_1, \ldots, \mathbf{a}_m) = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{b}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq N\}$, there exist $\lambda_1, \ldots, \lambda_m \in \mathbb{R}_{\geq 0}$ such that we have: $\mathbf{w}_i^t = \sum_{k=1}^m \lambda_k \mathbf{a}_k$.
6. Since $\mathbf{y} \in C$ and $C = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}_i^t \mathbf{x} \leq \mathbf{0},\ 1 \leq i \leq m\}$, we deduce
$$\mathbf{w}_i^t \mathbf{y} = \sum_{k=1}^m \lambda_k \mathbf{a}_k^t \mathbf{w}_i \leq 0. \quad (2.8)$$
7. In contradiction with $\mathbf{w}_i^t \mathbf{y} > 0$.

□

Decomposition theorem for polyhedron (1/6)

Theorem 21

Let $P \subseteq \mathbb{R}^d$. Then, P is a polyhedron if and only if $P = Q + C$ for some polytope Q and some polyhedral cone C.



Proof.

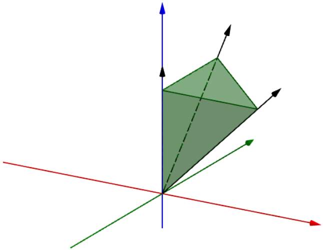
1. We will show that it follows from Farkas-Minkowsi-Weyl theorem.

□

Decomposition theorem for polyhedron (1/6)

Theorem 21

Let $P \subseteq \mathbb{R}^d$. Then, P is a polyhedron if and only if $P = Q + C$ for some polytope Q and some polyhedral cone C.



Proof.

1. We will show that it follows from Farkas-Minkowsi-Weyl theorem.
2. For this, we will rely on the geometric intuition that a polyhedron is a slice of a cone.

□

Decomposition theorem for polyhedron (2/6)

Proof.

1. Let $P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$ be a polyhedron.

□

Decomposition theorem for polyhedron (2/6)

Proof.

1. Let $P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$ be a polyhedron.
2. We are looking for a polytope Q and a cone C such that $P = Q + C$.

□

Decomposition theorem for polyhedron (2/6)

Proof.

1. Let $P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$ be a polyhedron.
2. We are looking for a polytope Q and a cone C such that $P = Q + C$.
   a. Define:

$$T = \left\{ \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} - \lambda \mathbf{b} \leq \mathbf{0}, \lambda \geq 0 \right\}.$$

□

Decomposition theorem for polyhedron (2/6)

Proof.

1. Let $P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \le \mathbf{b}\}$ be a polyhedron.
2. We are looking for a polytope Q and a cone C such that $P = Q + C$.
   - a. Define:
   $$T = \left\{ \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} - \lambda\mathbf{b} \le \mathbf{0}, \lambda \ge 0 \right\}.$$
   - b. Note that T is a polyhedral cone.

□

Decomposition theorem for polyhedron (2/6)

Proof.

1. Let $P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \le \mathbf{b}\}$ be a polyhedron.
2. We are looking for a polytope Q and a cone C such that $P = Q + C$.
   a. Define:
   $$T = \left\{ \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} - \lambda\mathbf{b} \le \mathbf{0}, \lambda \ge 0 \right\}.$$
   b. Note that T is a polyhedral cone.
   c. From the Farkas-Minkowsi-Weyl theorem, there exist $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ and $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ such hat
   $$T = \text{ConicalHull}(\begin{pmatrix} \mathbf{x}_1 \\ \lambda_1 \end{pmatrix}, \ldots, \begin{pmatrix} \mathbf{x}_m \\ \lambda_m \end{pmatrix}).$$

□

Decomposition theorem for polyhedron (2/6)

Proof.

1. Let $P = \{\mathbf{x} \in \mathbb{R}^d \mid A\mathbf{x} \leq \mathbf{b}\}$ be a polyhedron.
2. We are looking for a polytope Q and a cone C such that $P = Q + C$.
   a. Define:
   $$T = \left\{ \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} - \lambda\mathbf{b} \leq \mathbf{0}, \lambda \geq 0 \right\}.$$
   b. Note that T is a polyhedral cone.
   c. From the Farkas-Minkowsi-Weyl theorem, there exist $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ and $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ such hat
   $$T = \text{ConicalHull}(\begin{pmatrix} \mathbf{x}_1 \\ \lambda_1 \end{pmatrix}, \ldots, \begin{pmatrix} \mathbf{x}_m \\ \lambda_m \end{pmatrix}).$$
   d. By rescaling the elements of T, we can assume $\lambda_i \in \{0, 1\}$, for all i.

□

Decomposition theorem for polyhedron (3/6)

Proof.

Recall

$$T = \left\{ \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} - \lambda\mathbf{b} \leq \mathbf{0}, \lambda \geq 0 \right\} = \text{ConicalHull}(\begin{pmatrix} \mathbf{x}_1 \\ \lambda_1 \end{pmatrix}, \ldots, \begin{pmatrix} \mathbf{x}_m \\ \lambda_m \end{pmatrix}),$$

with $\lambda_i \in \{0, 1\}$, for all i.

□

Decomposition theorem for polyhedron (3/6)

Proof.

Recall

$$T = \left\{ \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} - \lambda \mathbf{b} \leq \mathbf{0}, \lambda \geq 0 \right\} = \text{ConicalHull}\left(\begin{pmatrix} \mathbf{x}_1 \\ \lambda_1 \end{pmatrix}, \ldots, \begin{pmatrix} \mathbf{x}_m \\ \lambda_m \end{pmatrix} \right),$$

with $\lambda_i \in \{0, 1\}$, for all i. For all $\mathbf{y} \in \mathbb{R}^d$ we have:

$$\begin{array}{rcl} \mathbf{y} \in P & \Longleftrightarrow & (\exists \lambda \in \mathbb{R}_{\geq 0}) \begin{pmatrix} \mathbf{y} \\ \lambda \end{pmatrix} \in T \\ & \Longleftrightarrow & (\exists \lambda, \gamma_1, \ldots, \gamma_m \in \mathbb{R}_{\geq 0}) \begin{pmatrix} \mathbf{y} \\ \lambda \end{pmatrix} = \gamma_1 \begin{pmatrix} \mathbf{x}_1 \\ \lambda_1 \end{pmatrix} + \cdots + \gamma_m \begin{pmatrix} \mathbf{x}_m \\ \lambda_m \end{pmatrix} \\ & \Longleftrightarrow & (\exists \gamma_1, \ldots, \gamma_m \in \mathbb{R}_{\geq 0}) \left\{ \begin{array}{l} \mathbf{y} = \sum_{i=1}^m \gamma_i \mathbf{x}_i \\ \sum_{i=1, \lambda_i = 1}^m \gamma_i = 1 \end{array} \right. \\ & \Longleftrightarrow & (\exists \gamma_1, \ldots, \gamma_m \in \mathbb{R}_{\geq 0}) \left\{ \begin{array}{l} \mathbf{y} = \sum_{i=1, \lambda_i = 0}^m \gamma_i \mathbf{x}_i + \sum_{i=1, \lambda_i = 1}^m \gamma_i \mathbf{x}_i \\ \sum_{i=1, \lambda_i = 1}^m \gamma_i = 1 \end{array} \right. \\ & \Longleftrightarrow & (\exists \gamma_1, \ldots, \gamma_m \in \mathbb{R}_{\geq 0}) \ \mathbf{y} \in C + Q, \end{array}$$

where:

$$C = \text{ConicalHull}(\mathbf{x}_i \mid \lambda_i = 0) \quad \text{and} \quad Q = \text{ConvexHull}(\mathbf{x}_i \mid \lambda_i = 1).$$

□

Decomposition theorem for polyhedron (4/6)

Proof.

1. Let $P = C + Q$, where C is a polyhedral cone and Q is a polytope.

□

Decomposition theorem for polyhedron (4/6)

Proof.

1. Let $P = C + Q$, where C is a polyhedral cone and Q is a polytope.
2. We need to show that P is a polyhedron.

□

Decomposition theorem for polyhedron (4/6)

Proof.

1. Let $P = C + Q$, where C is a polyhedral cone and Q is a polytope.
2. We need to show that P is a polyhedron.
3. From the Farkas-Minkowsi-Weyl theorem, there exist $\mathbf{r}_1, \ldots, \mathbf{r}_t \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{r}_1, \ldots, \mathbf{r}_t)$.

□

Decomposition theorem for polyhedron (4/6)

Proof.

1. Let $P = C + Q$, where C is a polyhedral cone and Q is a polytope.
2. We need to show that P is a polyhedron.
3. From the Farkas-Minkowsi-Weyl theorem, there exist $\mathbf{r}_1, \ldots, \mathbf{r}_t \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{r}_1, \ldots, \mathbf{r}_t)$.
4. By definition of a polytope, there exist $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $Q = \text{ConvexHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.

□

Decomposition theorem for polyhedron (4/6)

Proof.

1. Let $P = C + Q$, where C is a polyhedral cone and Q is a polytope.
2. We need to show that P is a polyhedron.
3. From the Farkas-Minkowsi-Weyl theorem, there exist $\mathbf{r}_1, \ldots, \mathbf{r}_t \in \mathbb{R}^d$ so that $C = \text{ConicalHull}(\mathbf{r}_1, \ldots, \mathbf{r}_t)$.
4. By definition of a polytope, there exist $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ so that $Q = \text{ConvexHull}(\mathbf{x}_1, \ldots, \mathbf{x}_m)$.
5. Then, we have:

$$\mathbf{y} \in P \iff (\exists \lambda_i, \gamma_j \in \mathbb{R}_{\geq 0}) \begin{cases} \mathbf{y} = \sum_{i=1}^{t} \lambda_i \mathbf{r}_i + \sum_{j=1}^{m} \gamma_j \mathbf{x}_j \\ \sum_{j=1}^{m} \gamma_j = 1 \end{cases}$$

$$\iff (\exists \lambda_i, \gamma_j \in \mathbb{R}_{\geq 0}) \begin{cases} \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} = \sum_{i=1}^{t} \lambda_i \begin{pmatrix} \mathbf{r}_i \\ 0 \end{pmatrix} + \sum_{j=1}^{m} \gamma_j \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix} \\ \sum_{j=1}^{m} \gamma_j = 1 \end{cases}$$

$$\iff \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} \in \text{ConicalHull}\left(\begin{pmatrix} \mathbf{r}_i \\ 0 \end{pmatrix}, \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix}, 1 \leq i \leq t, 1 \leq j \leq m\right)$$

□

Decomposition theorem for polyhedron (5/6)

Proof.

1. Denote by S the above cone, that is:

$$S = \text{ConicalHull}\left(\begin{pmatrix} \mathbf{r}_i \\ 0 \end{pmatrix}, \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix}, 1 \leq i \leq t, 1 \leq j \leq m\right).$$

□

Decomposition theorem for polyhedron (5/6)

Proof.

1. Denote by S the above cone, that is:

$$S = \text{ConicalHull}\left(\begin{pmatrix} \mathbf{r}_i \\ 0 \end{pmatrix}, \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix}, 1 \leq i \leq t, 1 \leq j \leq m\right).$$

2. Apply Farkas-Minkowski-Weyl theorem to S.

□

Decomposition theorem for polyhedron (5/6)

Proof.

1. Denote by S the above cone, that is:
$$S = \text{ConicalHull}(\begin{pmatrix} \mathbf{r}_i \\ 0 \end{pmatrix}, \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix}, 1 \leq i \leq t, 1 \leq j \leq m).$$
2. Apply Farkas-Minkowski-Weyl theorem to S.
3. Hence, there exists a matrix A and a vector $\mathbf{b}$ such that
$S = \{\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} + \lambda\mathbf{b} \leq 0\}$.

□

Decomposition theorem for polyhedron (5/6)

Proof.

1. Denote by S the above cone, that is:

$$S = \text{ConicalHull}\left(\begin{pmatrix} \mathbf{r}_i \\ 0 \end{pmatrix}, \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix}, 1 \le i \le t, 1 \le j \le m\right).$$

2. Apply Farkas-Minkowsi-Weyl theorem to S.
3. Hence, there exists a matrix A and a vector $\mathbf{b}$ such that $S = \{\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} + \lambda\mathbf{b} \le 0\}$.
4. Therefore, we have:

$$\begin{aligned} \mathbf{y} \in P &\iff \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} \in S \\ &\iff \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} \in \{\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} + \lambda\mathbf{b} \le 0\} \\ &\iff A\mathbf{y} \le -\mathbf{b}. \end{aligned}$$

□

Decomposition theorem for polyhedron (5/6)

Proof.

1. Denote by S the above cone, that is:
$$S = \text{ConicalHull}\left(\begin{pmatrix} \mathbf{r}_i \\ 0 \end{pmatrix}, \begin{pmatrix} \mathbf{x}_j \\ 1 \end{pmatrix}, 1 \le i \le t, 1 \le j \le m\right).$$
2. Apply Farkas-Minkowsi-Weyl theorem to S.
3. Hence, there exists a matrix A and a vector $\mathbf{b}$ such that $S = \{\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} + \lambda\mathbf{b} \le 0\}$.
4. Therefore, we have:
$$\begin{array}{rcl} \mathbf{y} \in P & \Longleftrightarrow & \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} \in S \\ & \Longleftrightarrow & \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} \in \{\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \mid A\mathbf{x} + \lambda\mathbf{b} \le 0\} \\ & \Longleftrightarrow & A\mathbf{y} \le -\mathbf{b}. \end{array}$$

This proves that P is a polyhedron and completes the proof. □

Decomposition theorem for polyhedron (6/6)

Corollary 22

Let $P \subseteq \mathbb{R}^d$. Then, P is a polytope if and only if it is a bounded polyhedron.

Proof.

1. Assume P is a polytope.

□

Decomposition theorem for polyhedron (6/6)

Corollary 22

Let $P \subseteq \mathbb{R}^d$. Then, P is a polytope if and only if it is a bounded polyhedron.

Proof.

1. Assume P is a polytope.
   a. Then, it is bounded as the it is the convex hull of finitely many points.

□

Decomposition theorem for polyhedron (6/6)

Corollary 22

Let $P \subseteq \mathbb{R}^d$. Then, P is a polytope if and only if it is a bounded polyhedron.

Proof.

1. Assume P is a polytope.
   a. Then, it is bounded as the it is the convex hull of finitely many points.
   b. Applying the decomposition theorem, and using the trivial cone (reduced to $\{\mathbf{0}\}$, P is also a polyhedron.

□

Decomposition theorem for polyhedron (6/6)

Corollary 22

Let $P \subseteq \mathbb{R}^d$. Then, P is a polytope if and only if it is a bounded polyhedron.

Proof.

1. Assume P is a polytope.
   - a. Then, it is bounded as the it is the convex hull of finitely many points.
   - b. Applying the decomposition theorem, and using the trivial cone (reduced to $\{\mathbf{0}\}$, P is also a polyhedron.
2. Assume P is a bounded polyhedron.

□

Decomposition theorem for polyhedron (6/6)

Corollary 22

Let $P \subseteq \mathbb{R}^d$. Then, P is a polytope if and only if it is a bounded polyhedron.

Proof.

1. Assume P is a polytope.
   a. Then, it is bounded as the it is the convex hull of finitely many points.
   b. Applying the decomposition theorem, and using the trivial cone (reduced to $\{\mathbf{0}\}$, P is also a polyhedron.
2. Assume P is a bounded polyhedron.
   a. Then, applying the decomposition theorem, and observing that the only bounded cone is $\{\mathbf{0}\}$, we deduce that P is a polytope.

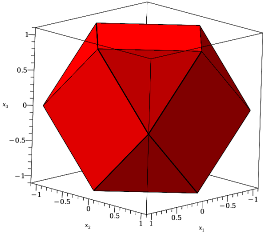
□

Plan

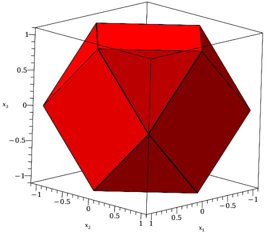
Plan

$$
\text{null}\begin{cases}
-x_3 \le 1 \\
-x_1 - x_2 - x_3 \le 2 \\
-x_1 + x_2 - x_3 \le 2 \\
x_1 - x_2 - x_3 \le 2 \\
x_1 + x_2 - x_3 \le 2 \\
x_3 0 \le 1 \\
-x_1 - x_2 + x_3 \le 2 \\
-x_1 + x_2 + x_3 \le 2 \\
x_1 - x_2 + x_3 \le 2 \\
x_1 + x_2 + x_3 \le 2 \\
-x_2 0 \le 1 \\
x_2 \le 1 \\
-x_1 \le 1 \\
x_1 0 \le 1
\end{cases}
$$

$$\left\{ \text{null} \begin{aligned} -x_3 &\le 1 \\ -x_1 - x_2 - x_3 &\le 2 \\ -x_1 + x_2 - x_3 &\le 2 \\ x_1 - x_2 - x_3 &\le 2 \\ x_1 + x_2 - x_3 &\le 2 \\ x_3 0 &\le 1 \\ -x_1 - x_2 + x_3 &\le 2 \\ -x_1 + x_2 + x_3 &\le 2 \\ x_1 - x_2 + x_3 &\le 2 \\ x_1 + x_2 + x_3 &\le 2 \\ -x_2 0 &\le 1 \\ x_2 &\le 1 \\ -x_1 &\le 1 \\ x_1 0 &\le 1 \end{aligned} \right.$$

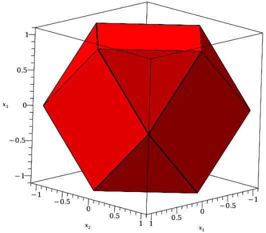


$$\text{null}\begin{cases} -x_3 \leq 1 \\ -x_1 - x_2 - x_3 \leq 2 \\ -x_1 + x_2 - x_3 \leq 2 \\ x_1 - x_2 - x_3 \leq 2 \\ x_1 + x_2 - x_3 \leq 2 \\ x_3 0 \leq 1 \\ -x_1 - x_2 + x_3 \leq 2 \\ -x_1 + x_2 + x_3 \leq 2 \\ x_1 - x_2 + x_3 \leq 2 \\ x_1 + x_2 + x_3 \leq 2 \\ -x_2 0 \leq 1 \\ x_2 \leq 1 \\ -x_1 \leq 1 \\ x_1 0 \leq 1 \end{cases}$$

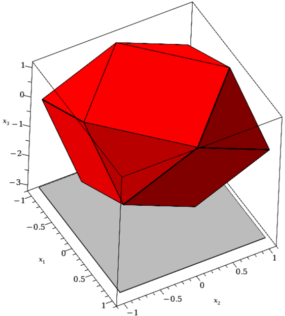


$$\text{null}\begin{cases} 0 \leq 1 + x_2 \\ 0 \leq 1 - x_2 \\ 0 \leq x_1 + 1 \\ 0 \leq 1 - x_1 \end{cases}$$

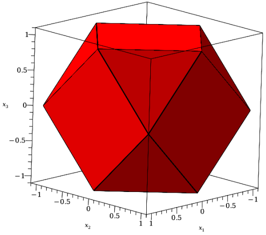
$$
\text{null}\begin{cases}
-x_3 \leq 1 \\
-x_1 - x_2 - x_3 \leq 2 \\
-x_1 + x_2 - x_3 \leq 2 \\
x_1 - x_2 - x_3 \leq 2 \\
x_1 + x_2 - x_3 \leq 2 \\
x_3 0 \leq 1 \\
-x_1 - x_2 + x_3 \leq 2 \\
-x_1 + x_2 + x_3 \leq 2 \\
x_1 - x_2 + x_3 \leq 2 \\
x_1 + x_2 + x_3 \leq 2 \\
-x_2 0 \leq 1 \\
x_2 \leq 1 \\
-x_1 \leq 1 \\
x_1 0 \leq 1
\end{cases}
$$



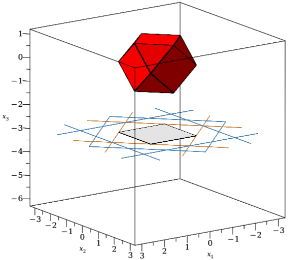
$$
\text{null}\begin{cases}
0 \leq 1 + x_2 \\
0 \leq 1 - x_2 \\
0 \leq x_1 + 1 \\
0 \leq 1 - x_1
\end{cases}
$$



$$
\text{null}\begin{cases}
-x_3 \leq 1 \\
-x_1 - x_2 - x_3 \leq 2 \\
-x_1 + x_2 - x_3 \leq 2 \\
x_1 - x_2 - x_3 \leq 2 \\
x_1 + x_2 - x_3 \leq 2 \\
x_3 0 \leq 1 \\
-x_1 - x_2 + x_3 \leq 2 \\
-x_1 + x_2 + x_3 \leq 2 \\
x_1 - x_2 + x_3 \leq 2 \\
x_1 + x_2 + x_3 \leq 2 \\
-x_2 0 \leq 1 \\
x_2 \leq 1 \\
-x_1 \leq 1 \\
x_1 0 \leq 1
\end{cases}
$$



$$
\text{null}\begin{cases}
0 \leq 1 + x_2 \\
0 \leq 1 - x_2 \\
0 \leq x_1 + 1 \\
0 \leq 1 - x_1
\end{cases}
$$



Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Application of FME: code generation

```
for(i=0; i<=n; i++){
   c[i] = 0; c[i+n] = 0;
   for(j=0; j<=n; j++)
      c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t, p) := (n - j, i + j)$.

Application of FME: code generation

```
for(i=0; i<=n; i++){
   c[i] = 0; c[i+n] = 0;
   for(j=0; j<=n; j++)
      c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t, p) := (n - j, i + j)$.

$$\text{null}\begin{cases} 0 \leq & i \\ i \leq & n \\ 0 \leq & j \\ j \leq & n \\ t = & n - j \\ p = & i + j \end{cases}$$

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t, p) := (n - j, i + j)$.

$$\text{null}\begin{cases} 0 \leq & i \\ i \leq & n \\ 0 \leq & j \\ j \leq & n \\ t = & n - j \\ p = & i + j \end{cases}$$

FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields:
$(t, p) := (n - j, i + j)$.

$$\begin{cases} 0 \leq & i \\ i \leq & n \\ 0 \leq & j \\ \text{null}\; j \leq & n \\ t = & n - j \\ p = & i + j \end{cases} \qquad \begin{cases} i = & p + t - n \\ j = & -t + n \\ t \geq & \max(0, -p + n) \\ \text{null}\; t \leq & \min(n, -p + 2n) \\ 0 \leq & p \\ p \leq & 2n \\ 0 \leq & n. \end{cases}$$

FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

Dependence analysis yields: $(t, p) := (n - j, i + j)$.

The new representation allows us to generate the multithreaded code.

$$\left\{ \begin{array}{rr} 0 \leq & i \\ i \leq & n \\ 0 \leq & j \\ j \leq & n \\ t = & n - j \\ p = & i + j \end{array} \right.$$

$$\left\{ \begin{array}{rr} i = & p + t - n \\ j = & -t + n \\ t \geq & \max(0, -p + n) \\ t \leq & \min(n, -p + 2n) \\ 0 \leq & p \\ p \leq & 2n \\ 0 \leq & n. \end{array} \right.$$

FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Application of FME: code generation

```
for(i=0; i<=n; i++){
  c[i] = 0; c[i+n] = 0;
  for(j=0; j<=n; j++)
    c[i+j] += a[i]*b[j];
}
```

```
parallel_for (p=0; p<=2*n; p++){
  c[p] = 0;
  for (t=max(0,n-p);
       t<=min(n,2*n-p);t++)
    c[p] += A[t+p-n] * B[n-t];
}
```

Dependence analysis yields: $(t, p) := (n - j, i + j)$.

The new representation allows us to generate the multithreaded code.

$$\left\{ \begin{array}{rr} 0 \leq & i \\ i \leq & n \\ 0 \leq & j \\ j \leq & n \\ t = & n - j \\ p = & i + j \end{array} \right.$$

$$\left\{ \begin{array}{rr} i = & p + t - n \\ j = & -t + n \\ t \geq & \max(0, -p + n) \\ t \leq & \min(n, -p + 2n) \\ 0 \leq & p \\ p \leq & 2n \\ 0 \leq & n. \end{array} \right.$$

FME reorders $p > t > i > j > n$ to $i > j > t > p > n$, thus eliminating i, j.

skip slide

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \leq \mathbf{b}$ has no implicit equation.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \le \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \le \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \le \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \leq \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.
6. From now, P is full-dimensional and pointed.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \le \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \le \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \le \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.
6. From now, P is full-dimensional and pointed.
7. Fixing $F : A\mathbf{x} \le \mathbf{b}$ an H-representation of P, a face of P is any intersection of P with the solution set of sub-system of F.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \leq \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.
6. From now, P is full-dimensional and pointed.
7. Fixing $F : A\mathbf{x} \leq \mathbf{b}$ an H-representation of P, a face of P is any intersection of P with the solution set of sub-system of F.
8. A vertex (resp. facet) is a face of dimension 0 (resp. $n-1$).

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \leq \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.
6. From now, P is full-dimensional and pointed.
7. Fixing $F : A\mathbf{x} \leq \mathbf{b}$ an H-representation of P, a face of P is any intersection of P with the solution set of sub-system of F.
8. A vertex (resp. facet) is a face of dimension 0 (resp. $n - 1$).
9. The characteristic cone of P is the polyhedral cone $\mathrm{CharCone}(P)$ represented by $\{A\mathbf{x} \leq \mathbf{0}\}$.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \leq \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.
6. From now, P is full-dimensional and pointed.
7. Fixing $F : A\mathbf{x} \leq \mathbf{b}$ an H-representation of P, a face of P is any intersection of P with the solution set of sub-system of F.
8. A vertex (resp. facet) is a face of dimension 0 (resp. $n - 1$).
9. The characteristic cone of P is the polyhedral cone $\mathrm{CharCone}(P)$ represented by $\{A\mathbf{x} \leq \mathbf{0}\}$.
10. Every polyhedral cone has a unique representation as a conical hull of its extremal generators, called the extreme rays of P.

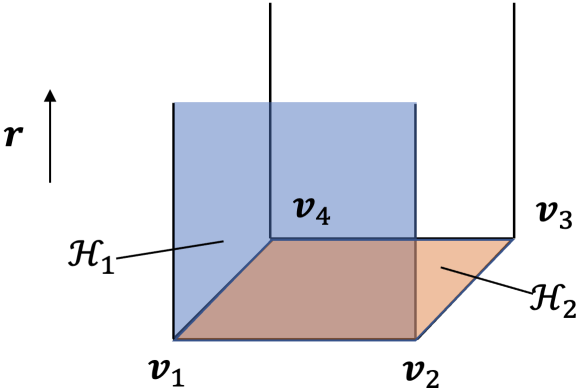
Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \leq \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.
6. From now, P is full-dimensional and pointed.
7. Fixing $F : A\mathbf{x} \leq \mathbf{b}$ an H-representation of P, a face of P is any intersection of P with the solution set of sub-system of F.
8. A vertex (resp. facet) is a face of dimension 0 (resp. $n - 1$).
9. The characteristic cone of P is the polyhedral cone $\mathrm{CharCone}(P)$ represented by $\{A\mathbf{x} \leq \mathbf{0}\}$.
10. Every polyhedral cone has a unique representation as a conical hull of its extremal generators, called the extreme rays of P.
11. Since P is pointed, an extreme ray of P is a one-dimensional face of $\mathrm{CharCone}(P)$.

Polyhedral sets

1. A polyhedral set $P \subseteq \mathbb{Q}^n$ is any $\{\mathbf{x} \in \mathbb{Q}^n \mid A\mathbf{x} \leq \mathbf{b}\}$, where $A \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Such a linear system is called an H-representation of P.
2. P is full-dimensional whenever $\dim(P) = n$
3. An inequality ℓ of $A\mathbf{x} \leq \mathbf{b}$ is an implicit equation if $\mathbf{a}^t\mathbf{x} = b$ holds for all $\mathbf{x} \in P$.
4. Thus, P is full-dimensional iff $A\mathbf{x} \leq \mathbf{b}$ has no implicit equation.
5. The polyhedron P is said pointed, if A is full column rank.
6. From now, P is full-dimensional and pointed.
7. Fixing $F : A\mathbf{x} \leq \mathbf{b}$ an H-representation of P, a face of P is any intersection of P with the solution set of sub-system of F.
8. A vertex (resp. facet) is a face of dimension 0 (resp. $n - 1$).
9. The characteristic cone of P is the polyhedral cone $\mathrm{CharCone}(P)$ represented by $\{A\mathbf{x} \leq \mathbf{0}\}$.
10. Every polyhedral cone has a unique representation as a conical hull of its extremal generators, called the extreme rays of P.
11. Since P is pointed, an extreme ray of P is a one-dimensional face of $\mathrm{CharCone}(P)$.
12. Let V and R denote the set of vertices and extreme rays of P. Then, the pair $\mathrm{Dual}(F) := (V, R)$ is called a V-representation of P.

An unbounded polyhedral set and its representations



The open cube $P := \{(x, y, z) \mid -z \leq 1, 0 \leq x \leq 1, 0 \leq y \leq 1\}$ shown above has 4 vertices $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4$ and extreme ray $\mathbf{r}$.

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

- redundant in F, if $F \setminus \{\ell\}$ still defines P,

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

- redundant in F, if $F \smallsetminus \{\ell\}$ still defines P,
- strongly redundant in F, if $\mathbf{a}^t\mathbf{x} < b$ holds for all $\mathbf{x} \in P$,

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

- redundant in F, if $F \smallsetminus \{\ell\}$ still defines P,
- strongly redundant in F, if $\mathbf{a}^t\mathbf{x} < b$ holds for all $\mathbf{x} \in P$,
- weakly redundant if it is redundant and $\mathbf{a}^t\mathbf{x} = b$ holds for some $\mathbf{x} \in P$.

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

- redundant in F, if $F \setminus \{\ell\}$ still defines P,
- strongly redundant in F, if $\mathbf{a}^t\mathbf{x} < b$ holds for all $\mathbf{x} \in P$,
- weakly redundant if it is redundant and $\mathbf{a}^t\mathbf{x} = b$ holds for some $\mathbf{x} \in P$.

Definition 24

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \le \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \le b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

- redundant in F, if $F \setminus \{\ell\}$ still defines P,
- strongly redundant in F, if $\mathbf{a}^t\mathbf{x} < b$ holds for all $\mathbf{x} \in P$,
- weakly redundant if it is redundant and $\mathbf{a}^t\mathbf{x} = b$ holds for some $\mathbf{x} \in P$.

Definition 24

- A vertex $v \in V$ of P saturates the inequality ℓ if $\mathbf{v}$ lies on $\mathcal{H}_\ell$, that is, if $\mathbf{a}^t\mathbf{v} = b$ holds.

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \leq \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \leq b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

- redundant in F, if $F \setminus \{\ell\}$ still defines P,
- strongly redundant in F, if $\mathbf{a}^t\mathbf{x} < b$ holds for all $\mathbf{x} \in P$,
- weakly redundant if it is redundant and $\mathbf{a}^t\mathbf{x} = b$ holds for some $\mathbf{x} \in P$.

Definition 24

- A vertex $v \in V$ of P saturates the inequality ℓ if $\mathbf{v}$ lies on $\mathcal{H}_\ell$, that is, if $\mathbf{a}^t\mathbf{v} = b$ holds.
- A ray $\mathbf{r} \in R$ of P saturates the inequality ℓ if $\mathbf{r}$ is parallel to the hyperplane $\mathcal{H}_\ell$, that is, if $\mathbf{a}^t\mathbf{r} = 0$ holds.

Redundant inequalities, the saturation matrix

1. Recall $F : A\mathbf{x} \le \mathbf{b}$ is an H-representation of our polyhedral set P.
2. Fix an inequality $\ell : \mathbf{a}^t\mathbf{x} \le b$ of F
3. Denote by $\mathcal{H}_\ell$ the hyperplane $\mathbf{a}^t\mathbf{x} = b$.
4. Recall V and R are the vertices and rays of P. Let $k := \#\mathrm{Dual}(F)$.

Definition 23

The inequality ℓ of F is

- redundant in F, if $F \setminus \{\ell\}$ still defines P,
- strongly redundant in F, if $\mathbf{a}^t\mathbf{x} < b$ holds for all $\mathbf{x} \in P$,
- weakly redundant if it is redundant and $\mathbf{a}^t\mathbf{x} = b$ holds for some $\mathbf{x} \in P$.

Definition 24

- A vertex $v \in V$ of P saturates the inequality ℓ if $\mathbf{v}$ lies on $\mathcal{H}_\ell$, that is, if $\mathbf{a}^t\mathbf{v} = b$ holds.
- A ray $\mathbf{r} \in R$ of P saturates the inequality ℓ if $\mathbf{r}$ is parallel to the hyperplane $\mathcal{H}_\ell$, that is, if $\mathbf{a}^t\mathbf{r} = 0$ holds.

The saturation matrix of F is the $0-1$ matrix $S \in \mathbb{Q}^{m\times k}$, where $S_{i,j} = 1$ iff the j-th element of $\mathrm{Dual}(F)$ saturates the i-th inequality of F.

A bounded polyhedral set and its the saturation matrix

F

$\ell_1 : x + y \leq 1$

null$\ell_2 : -x - y \leq 1$

$\ell_3 : x - y \leq 1$

$\ell_4 : -x + y \leq 1$

$\mathrm{Dual}(F)$

$\mathbf{v}_1 : (0, 1)$

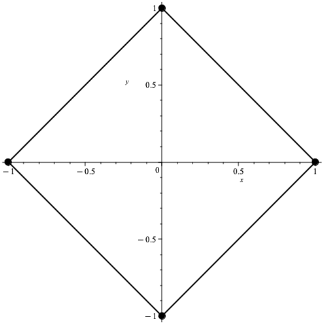
null$\mathbf{v}_2 : (1, 0)$

$\mathbf{v}_3 : (-1, 0)$

$\mathbf{v}_4 : (0, -1)$

satM(F)

	$\mathbf{v}_1$	$\mathbf{v}_2$	$\mathbf{v}_3$	$\mathbf{v}_4$
ℓ_1	1	1	0	0
ℓ_2	0	0	1	1
ℓ_3	0	1	0	1
ℓ_4	1	0	1	0



Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.

Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.
2. Write $\mathrm{Dual}(F) \cap \mathcal{H}_\ell := (\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}, \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s\})$, where $\mathbf{v}_i$'s are vertices and $\mathbf{r}_j$'s are rays.

Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.
2. Write $\mathrm{Dual}(F) \cap \mathcal{H}_\ell := (\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}, \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s\})$, where $\mathbf{v}_i$'s are vertices and $\mathbf{r}_j$'s are rays.
3. The affine rank of $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ is the rank of the matrix
$$[\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1, \ldots, \mathbf{v}_t - \mathbf{v}_1, \mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s].$$

Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.
2. Write $\mathrm{Dual}(F) \cap \mathcal{H}_\ell := (\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}, \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s\})$, where $\mathbf{v}_i$'s are vertices and $\mathbf{r}_j$'s are rays.
3. The affine rank of $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ is the rank of the matrix

$$[\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1, \ldots, \mathbf{v}_t - \mathbf{v}_1, \mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s].$$

With these notations, we have the following lemma. We note that any permutation $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t)$ would leave this result unchanged.

Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.
2. Write $\mathrm{Dual}(F) \cap \mathcal{H}_\ell := (\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}, \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s\})$, where $\mathbf{v}_i$'s are vertices and $\mathbf{r}_j$'s are rays.
3. The affine rank of $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ is the rank of the matrix

$$[\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1, \ldots, \mathbf{v}_t - \mathbf{v}_1, \mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s].$$

With these notations, we have the following lemma. We note that any permutation $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t)$ would leave this result unchanged.

Lemma

Assume the inequalities of F define hyperplanes that are pairwise different. Then, the following conditions are equivalent:

Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.
2. Write $\mathrm{Dual}(F) \cap \mathcal{H}_\ell := (\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}, \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s\})$, where $\mathbf{v}_i$'s are vertices and $\mathbf{r}_j$'s are rays.
3. The affine rank of $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ is the rank of the matrix
$$[\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1, \ldots, \mathbf{v}_t - \mathbf{v}_1, \mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s].$$

With these notations, we have the following lemma. We note that any permutation $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t)$ would leave this result unchanged.

Lemma

Assume the inequalities of F define hyperplanes that are pairwise different. Then, the following conditions are equivalent:

1. The inequality $\ell \in F$ is irredundant,

Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.
2. Write $\mathrm{Dual}(F) \cap \mathcal{H}_\ell := (\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}, \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s\})$, where $\mathbf{v}_i$'s are vertices and $\mathbf{r}_j$'s are rays.
3. The affine rank of $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ is the rank of the matrix
$$[\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1, \ldots, \mathbf{v}_t - \mathbf{v}_1, \mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s].$$

With these notations, we have the following lemma. We note that any permutation $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t)$ would leave this result unchanged.

Lemma

Assume the inequalities of F define hyperplanes that are pairwise different. Then, the following conditions are equivalent:

1. The inequality $\ell \in F$ is irredundant,
2. $\mathcal{H}_\ell \cap P$ is a facet of the polyhedron P.

Basic redundancy check

1. Denote by $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ the vertices and rays in $\mathrm{Dual}(F)$ saturating the hyperplane $\mathcal{H}_\ell$.
2. Write $\mathrm{Dual}(F) \cap \mathcal{H}_\ell := (\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t\}, \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s\})$, where $\mathbf{v}_i$'s are vertices and $\mathbf{r}_j$'s are rays.
3. The affine rank of $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ is the rank of the matrix

$$[\mathbf{v}_2 - \mathbf{v}_1, \mathbf{v}_3 - \mathbf{v}_1, \ldots, \mathbf{v}_t - \mathbf{v}_1, \mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_s].$$

With these notations, we have the following lemma. We note that any permutation $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t)$ would leave this result unchanged.

Lemma

Assume the inequalities of F define hyperplanes that are pairwise different. Then, the following conditions are equivalent:

1. The inequality $\ell \in F$ is irredundant,
2. $\mathcal{H}_\ell \cap P$ is a facet of the polyhedron P.
3. The affine rank of $\mathrm{Dual}(F) \cap \mathcal{H}_\ell$ equals to $n - 1$.

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.
2. For any ray or vertex $\mathbf{u}$, the set $\mathcal{S}^{\mathcal{H}}(\mathbf{u})$ collects all the hyperplanes saturated by $\mathbf{u}$.

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.
2. For any ray or vertex $\mathbf{u}$, the set $\mathcal{S}^{\mathcal{H}}(\mathbf{u})$ collects all the hyperplanes saturated by $\mathbf{u}$.
3. Fix an inequality ℓ of F.

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.
2. For any ray or vertex $\mathbf{u}$, the set $\mathcal{S}^{\mathcal{H}}(\mathbf{u})$ collects all the hyperplanes saturated by $\mathbf{u}$.
3. Fix an inequality ℓ of F.
4. Hence, the set
$$\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) := \bigcap_{\mathbf{u} \in \mathcal{S}^{\mathcal{VR}}(\ell)} \mathcal{S}^{\mathcal{H}}(\mathbf{u}),$$
is the set of all inequalities saturated by all the vertices or rays saturating ℓ.

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.
2. For any ray or vertex $\mathbf{u}$, the set $\mathcal{S}^{\mathcal{H}}(\mathbf{u})$ collects all the hyperplanes saturated by $\mathbf{u}$.
3. Fix an inequality ℓ of F.
4. Hence, the set
$$\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) := \bigcap_{\mathbf{u} \in \mathcal{S}^{\mathcal{VR}}(\ell)} \mathcal{S}^{\mathcal{H}}(\mathbf{u}),$$
is the set of all inequalities saturated by all the vertices or rays saturating ℓ.

Theorem 25

Let ℓ be an inequality in F. The following properties hold:

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.
2. For any ray or vertex $\mathbf{u}$, the set $\mathcal{S}^{\mathcal{H}}(\mathbf{u})$ collects all the hyperplanes saturated by $\mathbf{u}$.
3. Fix an inequality ℓ of F.
4. Hence, the set
$$\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) := \bigcap_{\mathbf{u} \in \mathcal{S}^{\mathcal{VR}}(\ell)} \mathcal{S}^{\mathcal{H}}(\mathbf{u}),$$
is the set of all inequalities saturated by all the vertices or rays saturating ℓ.

Theorem 25

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.
2. For any ray or vertex $\mathbf{u}$, the set $\mathcal{S}^{\mathcal{H}}(\mathbf{u})$ collects all the hyperplanes saturated by $\mathbf{u}$.
3. Fix an inequality ℓ of F.
4. Hence, the set
$$\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) := \bigcap_{\mathbf{u} \in \mathcal{S}^{\mathcal{VR}}(\ell)} \mathcal{S}^{\mathcal{H}}(\mathbf{u}),$$
is the set of all inequalities saturated by all the vertices or rays saturating ℓ.

Theorem 25

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*

Redundancy tests (1/2)

1. For any inequality ℓ, the set $\mathcal{S}^{\mathcal{VR}}(\ell)$ collects all the vertices and rays saturating ℓ.
2. For any ray or vertex $\mathbf{u}$, the set $\mathcal{S}^{\mathcal{H}}(\mathbf{u})$ collects all the hyperplanes saturated by $\mathbf{u}$.
3. Fix an inequality ℓ of F.
4. Hence, the set
$$\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) := \bigcap_{\mathbf{u}\in\mathcal{S}^{\mathcal{VR}}(\ell)} \mathcal{S}^{\mathcal{H}}(\mathbf{u}),$$
is the set of all inequalities saturated by all the vertices or rays saturating ℓ.

Theorem 25

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

Redundancy tests (2/2)

Theorem 26 (Recall from previous slide)

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

Redundancy tests (2/2)

Theorem 26 (Recall from previous slide)

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

- Denote by satM(F) the saturation matrix of F.

Redundancy tests (2/2)

Theorem 26 (Recall from previous slide)

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

- Denote by $\mathrm{satM}(F)$ the saturation matrix of F.
- $\mathrm{satM}(F)[\ell]$ is the row in $\mathrm{satM}(F)$ corresponding to ℓ, for $\ell \in F$.

Redundancy tests (2/2)

Theorem 26 (Recall from previous slide)

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

- Denote by $\mathrm{satM}(F)$ the saturation matrix of F.
- $\mathrm{satM}(F)[\ell]$ is the row in $\mathrm{satM}(F)$ corresponding to ℓ, for $\ell \in F$.

Corollary

The following properties hold:

Redundancy tests (2/2)

Theorem 26 (Recall from previous slide)

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

- Denote by satM(F) the saturation matrix of F.
- satM$(F)[\ell]$ is the row in satM(F) corresponding to ℓ, for $\ell \in F$.

Corollary

The following properties hold:

1. If satM$(F)[\ell]$ contains zeros only, then ℓ is strongly redundant.

Redundancy tests (2/2)

Theorem 26 (Recall from previous slide)

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

- Denote by $\mathrm{satM}(F)$ the saturation matrix of F.
- $\mathrm{satM}(F)[\ell]$ is the row in $\mathrm{satM}(F)$ corresponding to ℓ, for $\ell \in F$.

Corollary

The following properties hold:

1. If $\mathrm{satM}(F)[\ell]$ contains zeros only, then ℓ is strongly redundant.
2. If the number of nonzeros of $\mathrm{satM}(F)[\ell]$ is positive and less than the dimension n, then ℓ is weakly redundant.

Redundancy tests (2/2)

Theorem 26 (Recall from previous slide)

Let ℓ be an inequality in F. The following properties hold:

1. *The inequality ℓ is strongly redundant in F iff $\mathcal{S}^{\mathcal{VR}}(\ell)$ is empty.*
2. *If $\mathcal{S}^{\mathcal{VR}}(\ell)$ is non-empty and its cardinality is less than n, then the inequality ℓ is weakly redundant in F.*
3. *The inequality ℓ is weakly redundant in F iff the set $\mathcal{S}^{\mathcal{H}}(\mathcal{S}^{\mathcal{VR}}(\ell)) \setminus \{\ell\}$ is not empty.*

- Denote by $\mathrm{satM}(F)$ the saturation matrix of F.
- $\mathrm{satM}(F)[\ell]$ is the row in $\mathrm{satM}(F)$ corresponding to ℓ, for $\ell \in F$.

Corollary

The following properties hold:

1. If $\mathrm{satM}(F)[\ell]$ contains zeros only, then ℓ is strongly redundant.
2. If the number of nonzeros of $\mathrm{satM}(F)[\ell]$ is positive and less than the dimension n, then ℓ is weakly redundant.
3. If $\mathrm{satM}(F)[\ell]$ is contained in $\mathrm{satM}(F)[\ell_1]$ for some $\ell_1 \in F \setminus \{\ell\}$, then ℓ is weakly redundant.

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.

skip slide

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.
- Let $\ell_{pos} : a_1 x + \mathbf{c}_1^t \mathbf{y} \leq b_1$ and $\ell_{neg} : a_2 x + \mathbf{c}_2^t \mathbf{y} \leq b_2$, be two inequalities in x, where:

skip slide

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.
- Let $\ell_{pos} : a_1 x + \mathbf{c}_1^t \mathbf{y} \leq b_1$ and $\ell_{neg} : a_2 x + \mathbf{c}_2^t \mathbf{y} \leq b_2$, be two inequalities in x, where:
  1. we have $a_1 > 0$ and $a_2 < 0$,

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.
- Let $\ell_{pos} : a_1 x + \mathbf{c}_1^t \mathbf{y} \leq b_1$ and $\ell_{neg} : a_2 x + \mathbf{c}_2^t \mathbf{y} \leq b_2$, be two inequalities in x, where:
  1. we have $a_1 > 0$ and $a_2 < 0$,
  2. $\mathbf{y}$ is the vector of the remaining $(n - 1)$ variables, and

skip slide

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.
- Let $\ell_{pos} : a_1 x + \mathbf{c}_1^t \mathbf{y} \leq b_1$ and $\ell_{neg} : a_2 x + \mathbf{c}_2^t \mathbf{y} \leq b_2$, be two inequalities in x, where:
  1. we have $a_1 > 0$ and $a_2 < 0$,
  2. $\mathbf{y}$ is the vector of the remaining $(n-1)$ variables, and
  3. $\mathbf{c}_1, \mathbf{c}_2$ are the corresponding coefficient vectors.

skip slide

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.
- Let $\ell_{pos} : a_1 x + \mathbf{c}_1^t \mathbf{y} \leq b_1$ and $\ell_{neg} : a_2 x + \mathbf{c}_2^t \mathbf{y} \leq b_2$, be two inequalities in x, where:
  1. we have $a_1 > 0$ and $a_2 < 0$,
  2. $\mathbf{y}$ is the vector of the remaining $(n-1)$ variables, and
  3. $\mathbf{c}_1, \mathbf{c}_2$ are the corresponding coefficient vectors.
- Then, we have

$$\mathrm{proj}(\{\ell_{pos}, \ell_{neg}\}, \{x\}) = \{-a_2 \mathbf{c}_1^t \mathbf{y} + a_1 \mathbf{c}_2^t \mathbf{y} \leq -a_2 b_1 + a_1 b_2\}.$$

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.
- Let $\ell_{pos} : a_1 x + \mathbf{c}_1^t \mathbf{y} \leq b_1$ and $\ell_{neg} : a_2 x + \mathbf{c}_2^t \mathbf{y} \leq b_2$, be two inequalities in x, where:
  1. we have $a_1 > 0$ and $a_2 < 0$,
  2. $\mathbf{y}$ is the vector of the remaining $(n-1)$ variables, and
  3. $\mathbf{c}_1, \mathbf{c}_2$ are the corresponding coefficient vectors.
- Then, we have
$$\mathrm{proj}(\{\ell_{pos}, \ell_{neg}\}, \{x\}) = \{-a_2 \mathbf{c}_1^t \mathbf{y} + a_1 \mathbf{c}_2^t \mathbf{y} \leq -a_2 b_1 + a_1 b_2\}.$$

After computing all $\mathrm{proj}(\{\ell_{pos}, \ell_{neg}\}, \{x\})$'s and eliminating the redundant such inequalities, how to update the saturation matrix and prepare for the next variable elimination?

Updating satM(F) after eliminating one variable

- Consider the elimination of a variable, say x, during FME.
- Let $\ell_{pos} : a_1 x + \mathbf{c}_1^t \mathbf{y} \leq b_1$ and $\ell_{neg} : a_2 x + \mathbf{c}_2^t \mathbf{y} \leq b_2$, be two inequalities in x, where:
  1. we have $a_1 > 0$ and $a_2 < 0$,
  2. $\mathbf{y}$ is the vector of the remaining $(n-1)$ variables, and
  3. $\mathbf{c}_1, \mathbf{c}_2$ are the corresponding coefficient vectors.
- Then, we have
$$\mathrm{proj}(\{\ell_{pos}, \ell_{neg}\}, \{x\}) = \{-a_2 \mathbf{c}_1^t \mathbf{y} + a_1 \mathbf{c}_2^t \mathbf{y} \leq -a_2 b_1 + a_1 b_2\}.$$

After computing all $\mathrm{proj}(\{\ell_{pos}, \ell_{neg}\}, \{x\})$'s and eliminating the redundant such inequalities, how to update the saturation matrix and prepare for the next variable elimination?

Theorem 27

We have:

$$\mathcal{S}^{\mathcal{VR}}(\mathrm{proj}(\{\ell_{pos}, \ell_{neg}\}, \{x\})) = \mathrm{proj}(\mathcal{S}^{\mathcal{VR}}(\ell_{pos}) \cap \mathcal{S}^{\mathcal{VR}}(\ell_{neg}), \{x\}).$$

skip slide

Algorithm 1: CheckRedundancy

Input: 1. the inequality system F with m inequalities;
2. the saturation matrix satM.

Output: the minimal system F_{irred} and the corresponding saturation matrix $\mathrm{satM}_{\mathrm{irred}}$.

Irredundant := $\{\mathrm{seq}(i, i = 1..m)\}$.
for i from 1 to m **do**
    if the number of nonzero elements in $\mathrm{satM}[i]$ is less than n **then**
        Irredundant := *Irredundant* $\setminus \{i\}$.
        next.
    for j in *Irredundant* $\setminus \{i\}$ **do**
        if $\mathrm{satM}[i] = \mathrm{satM}[i] \& \mathrm{satM}[j]$ **then**
            Irredundant := *Irredundant* $\setminus \{i\}$.
            break.
F_{irred} := $[\mathrm{seq}(F[i], i \text{ in } \textit{Irredundant})]$ and
$\mathrm{satM}_{\mathrm{irred}}$:= $[\mathrm{seq}(\mathrm{satM}[i], i \text{ in } \textit{Irredundant})]$.
return F_{irred} and $\mathrm{satM}_{\mathrm{irred}}$.

Algorithm 2: Minimal projected representation

Input: 1. an inequality system F;
2. a variable order $x_1 > x_2 > \ldots > x_n$.

Output: the minimal projected representation *res* of F.

Compute the V-representation V of F by DD method;
Set $res :=$ table().
Sort the elements in V w.r.t. the reverse lexico order.
Compute the saturation matrix satM.
F, satM := $\mathrm{CheckRedundancy}(F, \mathrm{satM}(F))$.
$res[x_1] := F^{x_1}$.
for i from 1 to $n-1$ **do**
    $(F^p, F^n, F^0) :=$ partition(F).
    $V_{new} :=$ proj$(V, \{x_i\})$.
    Merging: satM := Merge(satM).
    Let $F_{new} := F^0$ and $\mathrm{satM}_{new} := \mathrm{satM}[F^0]$.
    foreach $f_p \in F^p$ and $f_n \in F^n$ **do**
        Append proj$((f_p, f_n), \{x_i\})$ to F_{new},
        Append $\mathrm{satM}[f_p]\&\mathrm{satM}[f_n]$ to satM_{new}.
    F, satM := $\mathrm{CheckRedundancy}(F_{new}, \mathrm{satM}_{new})$.
    $V := V_{new}$, $res[x_{i+1}] := F^{x_{i+1}}$.
return *res*.

Plan

Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).

Implementation techniques

1. Clearly, $\mathsf{satM}(F)$ should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.

Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both

Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both
    - row-wise (to compute bit-wise AND) Line 7 in Algorithm 1, and

Implementation techniques

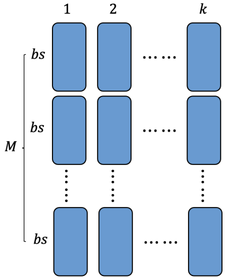
1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both
   - row-wise (to compute bit-wise AND) Line 7 in Algorithm 1, and
   - column-wise (to compute bit-wise OR) Line 10 in Algorithm 2.

Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both
   - row-wise (to compute bit-wise AND) Line 7 in Algorithm 1, and
   - column-wise (to compute bit-wise OR) Line 10 in Algorithm 2.
4. For cache complexity reasons, we maintain both satM(F) and satM$(F)^t$.

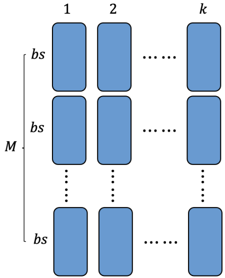
Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both
   - row-wise (to compute bit-wise AND) Line 7 in Algorithm 1, and
   - column-wise (to compute bit-wise OR) Line 10 in Algorithm 2.
4. For cache complexity reasons, we maintain both satM(F) and satM(F)^t.
5. Moreover, these matrices should be represented by blocks.



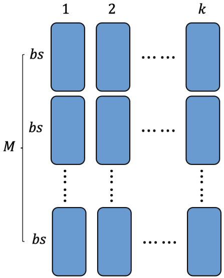
Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both
   - row-wise (to compute bit-wise AND) Line 7 in Algorithm 1, and
   - column-wise (to compute bit-wise OR) Line 10 in Algorithm 2.
4. For cache complexity reasons, we maintain both satM(F) and satM(F)^t.
5. Moreover, these matrices should be represented by blocks.
6. Other key tasks Algorithm 2 are



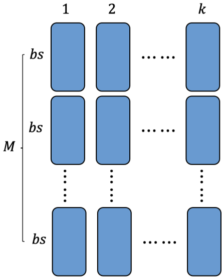
Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both
   - row-wise (to compute bit-wise AND) Line 7 in Algorithm 1, and
   - column-wise (to compute bit-wise OR) Line 10 in Algorithm 2.
4. For cache complexity reasons, we maintain both satM(F) and satM$(F)^t$.
5. Moreover, these matrices should be represented by blocks.
6. Other key tasks Algorithm 2 are
   - computing the V-representation of each successive projection



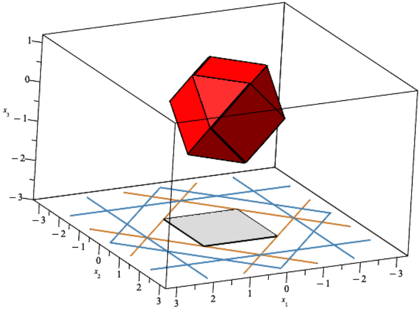
Implementation techniques

1. Clearly, satM(F) should be encoded with bit vectors (aka bit-arrays).
2. We use bitarray, the bitarray library by Michael Dipperstein.
3. satM(F) is traversed both
   - row-wise (to compute bit-wise AND) Line 7 in Algorithm 1, and
   - column-wise (to compute bit-wise OR) Line 10 in Algorithm 2.
4. For cache complexity reasons, we maintain both satM(F) and satM$(F)^t$.
5. Moreover, these matrices should be represented by blocks.
6. Other key tasks Algorithm 2 are
   - computing the V-representation of each successive projection
   - updating the saturation matrix.



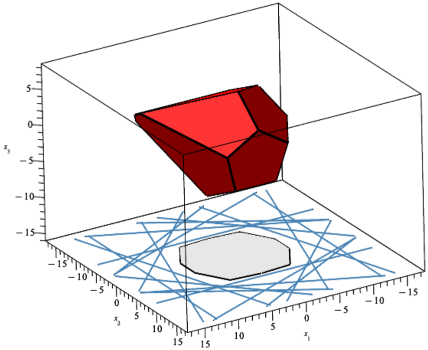
Plan

Cuboctahedron



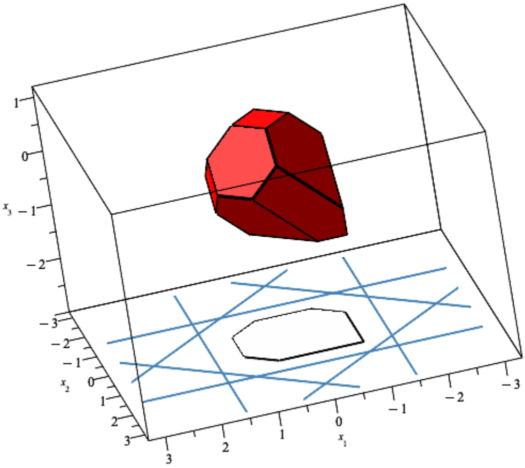
1. **strongly redundannt inequalities**
2. **weakly redundant inequalities eliminated by cardinality**
3. **weakly redundancies inequalities eliminated by containment**

Snub disphenoid (triangular dodecahedron)

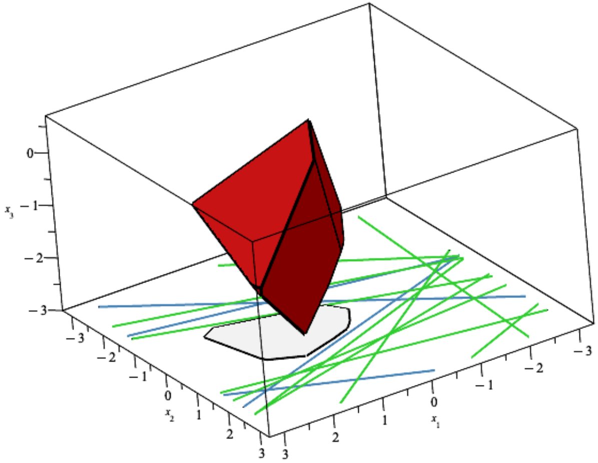


1. **strongly redundannt inequalities**
2. **weakly redundant inequalities eliminated by cardinality**
3. **weakly redundancies inequalities eliminated by containment**

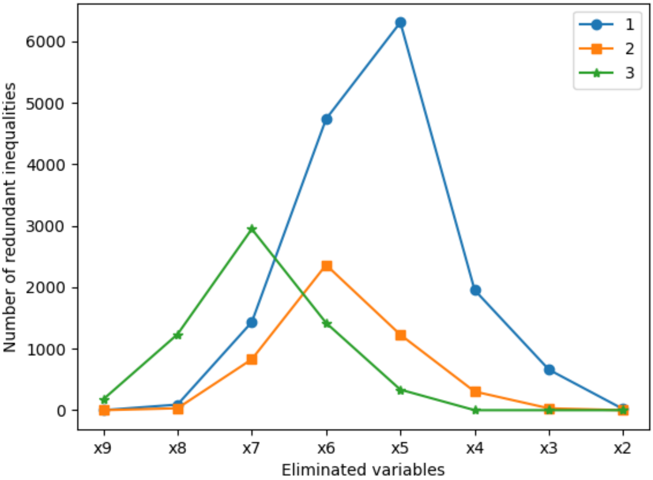
Truncated octahedron



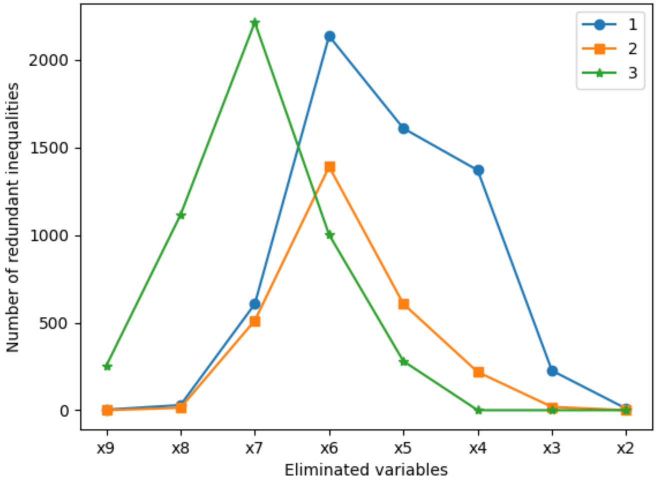
Random 3D polyhedron



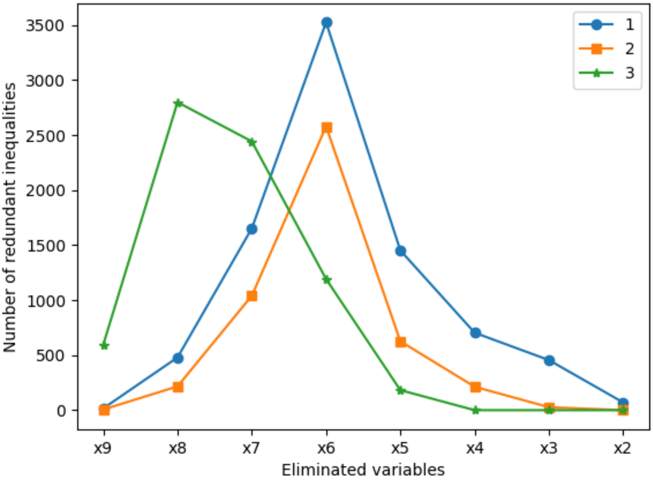
Random 10D polyhedron



Random 10D polyhedron



Random 10D polyhedron



Comparative experimentation (1/3)

Four ways of eliminating all variables:

- MPR (this paper): one variable after another, uses both the H-representation and V-representations, redundancy test via saturation matrices

Comparative experimentation (1/3)

Four ways of eliminating all variables:

- MPR (this paper): one variable after another, uses both the H-representation and V-representations, redundancy test via saturation matrices
- BPAS ([9] by Authors 1 and 2, with Delaram Talaashrafi): one variable after another, uses both the H-representation and V-representations, redundancy test via redundancy test cones, thus linear algebra over $\mathbb{Q}$.

Comparative experimentation (1/3)

Four ways of eliminating all variables:

- MPR (this paper): one variable after another, uses both the H-representation and V-representations, redundancy test via saturation matrices
- BPAS ([9] by Authors 1 and 2, with Delaram Talaashrafi): one variable after another, uses both the H-representation and V-representations, redundancy test via redundancy test cones, thus linear algebra over $\mathbb{Q}$.
- cddlib [5] by Komei Fukuda: can eliminate several variables in one step, can work with the H-representation only, redundancy test via Linear Programming (LP).

Comparative experimentation (1/3)

Four ways of eliminating all variables:

- MPR (this paper): one variable after another, uses both the H-representation and V-representations, redundancy test via saturation matrices
- BPAS ([9] by Authors 1 and 2, with Delaram Talaashrafi): one variable after another, uses both the H-representation and V-representations, redundancy test via redundancy test cones, thus linear algebra over $\mathbb{Q}$.
- cddlib [5] by Komei Fukuda: can eliminate several variables in one step, can work with the H-representation only, redundancy test via Linear Programming (LP).
- polylib [13] by Vincent Loechner and Doran K. Wilde: can eliminate several variables in one step, can work with the V-representation only, convert between H-rep and V-rep as needed.

Comparative experimentation (1/3)

Four ways of eliminating all variables:

- MPR (this paper): one variable after another, uses both the H-representation and V-representations, redundancy test via saturation matrices
- BPAS ([9] by Authors 1 and 2, with Delaram Talaashrafi): one variable after another, uses both the H-representation and V-representations, redundancy test via redundancy test cones, thus linear algebra over $\mathbb{Q}$.
- cddlib [5] by Komei Fukuda: can eliminate several variables in one step, can work with the H-representation only, redundancy test via Linear Programming (LP).
- polylib [13] by Vincent Loechner and Doran K. Wilde: can eliminate several variables in one step, can work with the V-representation only, convert between H-rep and V-rep as needed.

We used the following sources for our test cases:

1. random non-empty polyhedra with n variables and m inequalities. The coefficients rang in the interval $[-10, 10]$.
2. polyhedra coming from libraries polylib and BPAS.

Comparative experimentation (1/3)

Four ways of eliminating all variables:

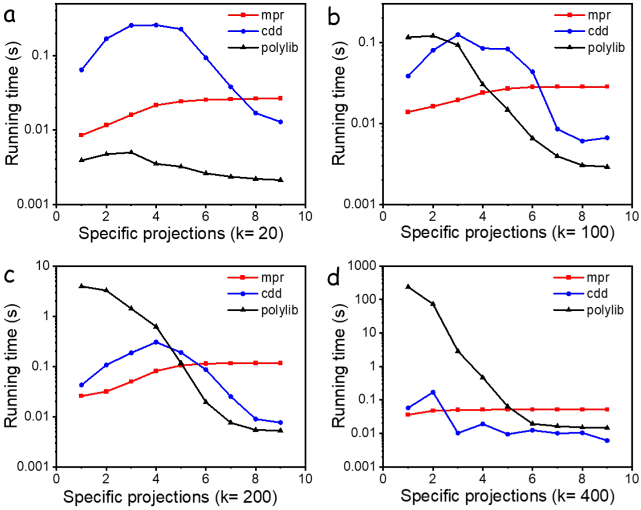
- MPR (this paper): one variable after another, uses both the H-representation and V-representations, redundancy test via saturation matrices
- BPAS ([9] by Authors 1 and 2, with Delaram Talaashrafi): one variable after another, uses both the H-representation and V-representations, redundancy test via redundancy test cones, thus linear algebra over $\mathbb{Q}$.
- cddlib [5] by Komei Fukuda: can eliminate several variables in one step, can work with the H-representation only, redundancy test via Linear Programming (LP).
- polylib [13] by Vincent Loechner and Doran K. Wilde: can eliminate several variables in one step, can work with the V-representation only, convert between H-rep and V-rep as needed.

We used the following sources for our test cases:

1. random non-empty polyhedra with n variables and m inequalities. The coefficients rang in the interval $[-10, 10]$.
2. polyhedra coming from libraries polylib and BPAS.

All the experimental results were collected on a PC (Intel(R) Xeon(R) Gold 6258R CPU 2.70GHz, 503G RAM, Ubuntu 20.04.3).

Comparative experimentation (2/3)



1. Four different random polyhedra with $m = 15$ and $n = 10$.
2. For $1 \leq i \leq 9$, in the hor. axiss, the first i variables are eliminated.
3. The vert. axis in each figure shows the running time (in seconds).

test case	(n, m, k)	mpr (msec.)	BPAS (msec.)	cdd (msec.)	polylib (msec.)
32hedron	(6, 32, 11)	6.54	16.80	4183.08	**1.92**
64hedron	(7,64,13)	13.05	52.42	>5min	**1.67**
francois	(13,27,2304)	499.92	**253.66**	388.36	> 5min
francois2	(13,31,384)	**41.80**	140.34	55.17	80.63
herve.in	(14,25,262)	34.42	140.34	294.01	**30.08**
c6.in	(11,17,31)	**9.85**	12.72	84.11	5.56
c9.in	(16,18,140)	**25.08**	65.54	151.17	131.53
c10.in	(18,20,142)	22.10	98.68	249.02	**16.06**
S24	(24, 25,25)	23.50	58.80	748.67	**17.47**
S35	(35, 36,36)	46.55	182.14	3575.00	**46.007**
cube	(10, 20,1024)	**81.33**	201.92	125.900	161.06
C56	(5, 6,6)	3.67	4.09	11.81	**0.79**
C1011	(10, 11,11)	24.99	115.68	1716.25	**9.99**
C510	(5, 42,10)	12.00	40.01	>5min	**4.42**
T1	(5, 10,38)	**5.61**	16.44	27.42	8.81
T3	(10,12,29)	21.29	141.64	288.07	**12.07**
T5	(5, 10,36)	8.12	15.62	22.92	**4.76**
T6	(10,20,390)	**1142.9**	23800.11	14937.61	>5min
T7	(5, 8,26)	5.81	10.79	13.96	**4.00**
T9	(10,12,36)	**36.56**	414.53	479.18	100.34
T10	(6, 8,24)	**4.58**	13.65	18.39	5.27
T12	(5, 11,42)	**8.52**	19.03	38.65	8.60
R_15_20	(15, 20,1328)	**28430.40**	336035.00	38037.21	>5min

Complexity estimates (1/2)

Recall the notations

1. m is the number of inequalities and n is the dimension of the ambient space. If the input H-representation is irredundant, the m is also the number of facets of P.
2. Let $h := \text{height}([A, \mathbf{b}])$, let θ be the coefficient of linear algebra and ω the bit-size of a machine word.

Well-known bounds

1. The size k of the V-representation (V, R) is at most $\binom{m}{n} + \binom{m}{n-1} \leq \frac{m^n}{n!}$.
2. From [8, 9] for $1 \leq i < n$, after eliminating i variables during the process of FME, the number of irredundant inequalities defining the projection is at most $\binom{m}{n-i-1} \leq m^n$.

Theorem 28

The costs for computing all the inequalities (redundant and irredundant) and generating the initial saturation matrix are within $O(m^{2n} n^{\theta+\varepsilon} h^{1+\varepsilon})$ bit operations, while the costs for updating and operating on the saturation matrices are bounded over by $\frac{3m^{3n-4}}{\omega}$ word operations.

Complexity estimates (1/2)

Recall the notations

1. m is the number of inequalities and n is the dimension of the ambient space. If the input H-representation is irredundant, the m is also the number of facets of P.
2. Let $h := \text{height}([A, \mathbf{b}])$, let θ be the coefficient of linear algebra and ω the bit-size of a machine word.

Bounds for FME

1. FME based on LP: $O(n^2\, m^{2n}\, \text{LP}(n, 2^n h n^2 m^n))$ bit operations, where $\text{LP}(d, H)$ is an upper bound for the number of bit operations required for solving a linear program in d variables and with total bit size H. For instance, in the case of Karmarkar's algorithm [12], we have $\text{LP}(d, H) \in O(d^{3.5} H^2 \cdot \log H \cdot \log\log H)$.
2. FME based on redundancy test cone: $O(m^{\frac{5n}{2}}\, n^{\theta+1+\epsilon} h^{1+\epsilon})$ bit operations, for any $\epsilon > 0$.
3. This paper: $O(m^{2n} n^{\theta+\varepsilon} h^{1+\varepsilon})$ bit operations and $\frac{3m^{3n-4}}{\omega}$ word operations.

Concluding remarks

Summary and notes

1. We proposed a technique for removing redundant inequalities in linear systems.
2. It relies on the analysis of 3 different types of redundancies
3. Our redundancy tests allow for efficient implementation based on bit-vector arithmetic.
4. From the experimental results, our method works best on hard problems.
5. This is promising to solve large scale problems in areas like information theory, SMT and optimizing compilers.

Work in progress

1. Our implementation has room for improvements.
2. Indeed, our algorithms have opportunities for both multithreaded parallelism and instruction-level parallelism.
3. The third criterion (redundancy test based on containment) needs further study to discover the container.

Plan

Plan

Dependence analysis

Cholesky's LU decomposition:

```
1:   for(i = 1; i <= n; i++){
        x = a[i][i];
        for(k = 1; k < i; k++)
2:          x = x - a[i][k] * a[i][k];
3:      p[i] = 1.0/sqrt(x);
        for(j = i + 1; j <= n; j++){
4:          x = a[i][j];
            for(k = 1; k < i; k++)
5:              x = x - a[j][k] * a[i][k];
6:          a[j][i] = x * p[i];
        }
     }
```

Dependence analysis

Cholesky's LU decomposition:

```
1:   for(i = 1; i <= n; i++){
         x = a[i][i];
         for(k = 1; k < i; k++)
2:           x = x - a[i][k] * a[i][k];
3:       p[i] = 1.0/sqrt(x);
         for(j = i + 1; j <= n; j++){
4:           x = a[i][j];
             for(k = 1; k < i; k++)
5:               x = x - a[j][k] * a[i][k];
6:           a[j][i] = x * p[i];
         }
     }
```

system 1:

$$\left\{\begin{array}{lr} & 1 \le i \le n \\ & i+1 \le j \le n \\ & 1 \le k \le i-1 \\ \text{null} & 1 \le i' \le n \\ & i'+1 \le j' \le n \\ & j = j', k = i' \\ & i < i' \end{array}\right.$$

system 2:

$$\left\{\begin{array}{lr} & 1 \le i \le n \\ & i+1 \le j \le n \\ & 1 \le k \le i-1 \\ \text{null} & 1 \le i' \le n \\ & i'+1 \le j' \le n \\ & j = j', k = i' \\ & i = i', j < j' \end{array}\right.$$

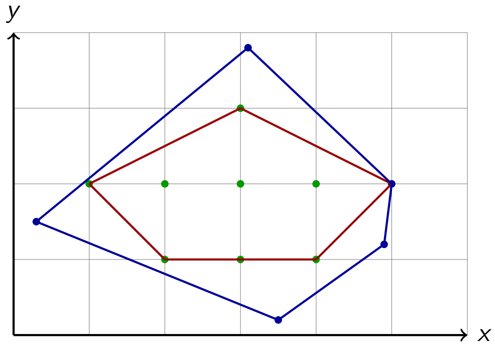
system 3:

$$\left\{\begin{array}{lr} & 1 \le i \le n \\ & i+1 \le j \le n \\ & 1 \le k \le i-1 \\ \text{null} & 1 \le i' \le n \\ & i'+1 \le j' \le n \\ & j = j', k = i' \\ & i = i', j = j' \end{array}\right.$$

Plan

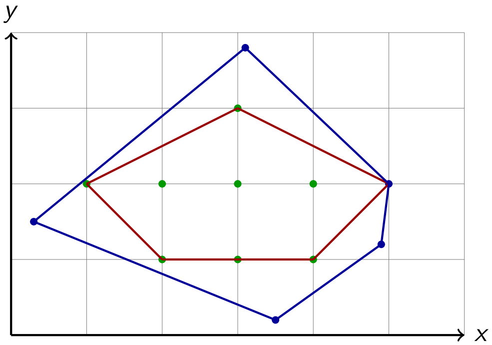
Integer Hull

1. The **integer hull** of the polyhedron $P \subseteq \mathbb{Q}^d$, denoted by P_I, is the smallest convex polyhedron containing all the integer points of P.



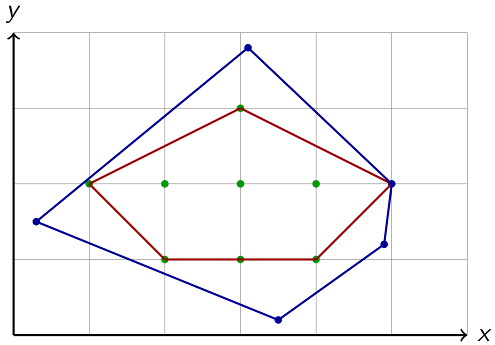
Integer Hull

1. The **integer hull** of the polyhedron $P \subseteq \mathbb{Q}^d$, denoted by P_I, is the smallest convex polyhedron containing all the integer points of P.
2. In other words, P_I is the intersection of all convex polyhedra containing $P \cap \mathbb{Z}^d$.



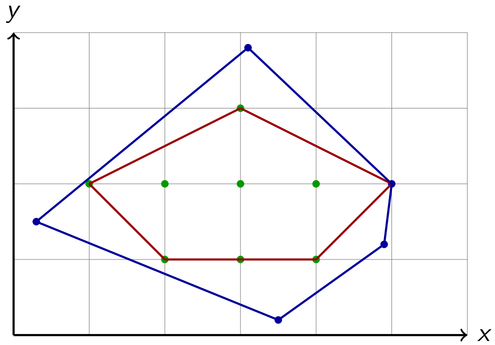
Integer Hull

1. The **integer hull** of the polyhedron $P \subseteq \mathbb{Q}^d$, denoted by P_I, is the smallest convex polyhedron containing all the integer points of P.
2. In other words, P_I is the intersection of all convex polyhedra containing $P \cap \mathbb{Z}^d$.
3. Assume that P is pointed (that is, $A\mathbf{x} = \mathbf{0} \Rightarrow \mathbf{x} = \mathbf{0}$, for $P = \text{Polyhedron}(A, \mathbf{b})$). Then, $P = P_I$ holds if and only if every vertex of P is integral.



Integer Hull

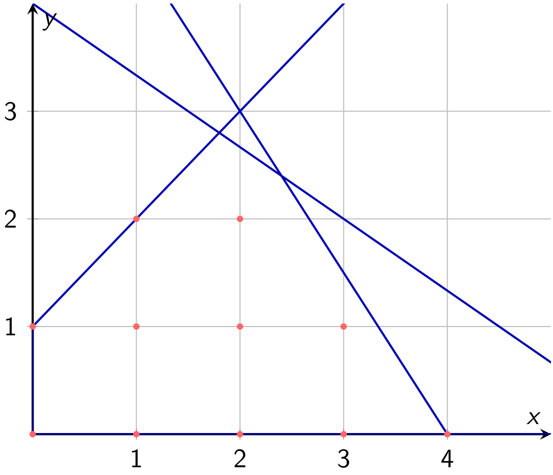
1. The **integer hull** of the polyhedron $P \subseteq \mathbb{Q}^d$, denoted by P_I, is the smallest convex polyhedron containing all the integer points of P.
2. In other words, P_I is the intersection of all convex polyhedra containing $P \cap \mathbb{Z}^d$.
3. Assume that P is pointed (that is, $A\mathbf{x} = \mathbf{0} \Rightarrow \mathbf{x} = \mathbf{0}$, for $P = \text{Polyhedron}(A, \mathbf{b})$). Then, $P = P_I$ holds if and only if every vertex of P is integral.
4. Thus, for P pointed, the convex hull of all the vertices of P_I is P_I itself.



Integer hull: simple example

$$\begin{cases} 0 & \leq & x \\ 0 & \leq & y \\ 3\,x + 2\,y & \leq & 12 \\ 2\,x + 3\,y & \leq & 12 \\ -x + y & \leq & 1 \end{cases}$$

$$\begin{cases} 0 & \leq & x \\ 0 & \leq & y \\ y & \leq & 2 \\ x + y & \leq & 4 \\ -x + y & \leq & 1 \end{cases}$$



Figure

Lattices

1. A subset $L \subseteq \mathbb{Z}^d$ is called an **integer lattice** (or simply a lattice) if
$$L = \{\mathbf{x} \in \mathbb{Z}^d \mid (\exists \mathbf{t} \in \mathbb{Z}^c)\ \mathbf{x} = A\mathbf{t} + \mathbf{b}\}$$
holds, for a matrix $A \in \mathbb{Z}^{d \times c}$ and a vector $\mathbf{b} \in \mathbb{Z}^d$, where c is a positive integer.

Lattices

1. A subset $L \subseteq \mathbb{Z}^d$ is called an **integer lattice** (or simply a lattice) if
$$L = \{\mathbf{x} \in \mathbb{Z}^d \mid (\exists \mathbf{t} \in \mathbb{Z}^c)\ \mathbf{x} = A\mathbf{t} + \mathbf{b}\}$$
holds, for a matrix $A \in \mathbb{Z}^{d \times c}$ and a vector $\mathbf{b} \in \mathbb{Z}^d$, where c is a positive integer.
2. It is convenient to see this lattice as the solution set of the systems of congruence relations
$$\mathbf{x} \equiv \mathbf{b} \mod A.$$

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where
   a. $\mathbf{0} \in \mathbb{Z}^{r \times (d-r)}$ is the null matrix, and

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where
   a. $\mathbf{0} \in \mathbb{Z}^{r \times (d-r)}$ is the null matrix, and
   b. H is the **column-style Hermite normal form** of C.

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^{r}$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where
   a. $\mathbf{0} \in \mathbb{Z}^{r \times (d-r)}$ is the null matrix, and
   b. H is the **column-style Hermite normal form** of C.
4. We write $U = [U_L U_R]$ where $U_L \in \mathbb{Z}^{d \times (d-r)}$ and $U_R \in \mathbb{Z}^{d \times r}$.

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^{r}$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where
    a. $\mathbf{0} \in \mathbb{Z}^{r \times (d-r)}$ is the null matrix, and
    b. H is the **column-style Hermite normal form** of C.
4. We write $U = [U_L U_R]$ where $U_L \in \mathbb{Z}^{d \times (d-r)}$ and $U_R \in \mathbb{Z}^{d \times r}$.
5. Therefore, the matrix $H \in \mathbb{Z}^{r \times r}$ is non-singular and the following properties hold:

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where
   a. $\mathbf{0} \in \mathbb{Z}^{r \times (d-r)}$ is the null matrix, and
   b. H is the **column-style Hermite normal form** of C.
4. We write $U = [U_L U_R]$ where $U_L \in \mathbb{Z}^{d \times (d-r)}$ and $U_R \in \mathbb{Z}^{d \times r}$.
5. Therefore, the matrix $H \in \mathbb{Z}^{r \times r}$ is non-singular and the following properties hold:
   a. $\{\mathbf{x} \in \mathbb{Z}^d \mid C\mathbf{x} = \mathbf{q}\} \neq \emptyset \iff H^{-1}\mathbf{q} \in \mathbb{Z}^r$,

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \le d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where
   a. $\mathbf{0} \in \mathbb{Z}^{r \times (d-r)}$ is the null matrix, and
   b. H is the **column-style Hermite normal form** of C.
4. We write $U = [U_L U_R]$ where $U_L \in \mathbb{Z}^{d \times (d-r)}$ and $U_R \in \mathbb{Z}^{d \times r}$.
5. Therefore, the matrix $H \in \mathbb{Z}^{r \times r}$ is non-singular and the following properties hold:
   a. $\{\mathbf{x} \in \mathbb{Z}^d \mid C\mathbf{x} = \mathbf{q}\} \neq \emptyset \iff H^{-1}\mathbf{q} \in \mathbb{Z}^r$,
   b. $\{\mathbf{x} \in \mathbb{Z}^d \mid C\mathbf{x} = \mathbf{q}\} = \{U_R H^{-1}\mathbf{q} + U_L\mathbf{v} \mid \mathbf{v} \in \mathbb{Z}^{d-r}\}$.

Hermite normal forms

1. Let $C \in \mathbb{Z}^{r \times d}$ and $\mathbf{q} \in \mathbb{Z}^r$, with $r \in \mathbb{Z}_{>0}$ and $r \leq d$.
2. Assume that C is a full row-rank matrix, thus the rank of C is r.
3. Then, there exists a unimodular matrix $U \in \mathbb{Z}^{d \times d}$ so that $CU = [\mathbf{0} H]$ where
   - a. $\mathbf{0} \in \mathbb{Z}^{r \times (d-r)}$ is the null matrix, and
   - b. H is the **column-style Hermite normal form** of C.
4. We write $U = [U_L U_R]$ where $U_L \in \mathbb{Z}^{d \times (d-r)}$ and $U_R \in \mathbb{Z}^{d \times r}$.
5. Therefore, the matrix $H \in \mathbb{Z}^{r \times r}$ is non-singular and the following properties hold:
   - a. $\{\mathbf{x} \in \mathbb{Z}^d \mid C\mathbf{x} = \mathbf{q}\} \neq \emptyset \iff H^{-1}\mathbf{q} \in \mathbb{Z}^r$,
   - b. $\{\mathbf{x} \in \mathbb{Z}^d \mid C\mathbf{x} = \mathbf{q}\} = \{U_R H^{-1}\mathbf{q} + U_L\mathbf{v} \mid \mathbf{v} \in \mathbb{Z}^{d-r}\}$.
6. These results generalize to the case where the rank of C is arbitrary, see [21],

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\text{Polyhedron}(P, L)$.

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\text{Polyhedron}(P, L)$.
2. This is a subset of the integer points of P, which can be empty.

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\text{Polyhedron}(P, L)$.
2. This is a subset of the integer points of P, which can be empty.
3. Denote by $x_1 < x_2 < \cdots < x_d$ the coordinates of $\mathbb{Z}^d$. We say that $\mathbb{Z}\text{Polyhedron}(P, L)$ is **normalized** if

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\text{Polyhedron}(P, L)$.
2. This is a subset of the integer points of P, which can be empty.
3. Denote by $x_1 < x_2 < \cdots < x_d$ the coordinates of $\mathbb{Z}^d$. We say that $\mathbb{Z}\text{Polyhedron}(P, L)$ is **normalized** if
   a. it is non-empty, and P is given by a system of linear inequalities of the form

$$\begin{cases} a_0 & \leq x_1 \leq & b_0 \\ a_1 & \leq x_2 \leq & b_1 \\ \vdots & \vdots & \vdots \\ a_{n-1} & \leq x_d \leq & b_{n-1}, \end{cases} \tag{4.1}$$

   where

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\mathrm{Polyhedron}(P, L)$.
2. This is a subset of the integer points of P, which can be empty.
3. Denote by $x_1 < x_2 < \cdots < x_d$ the coordinates of $\mathbb{Z}^d$. We say that $\mathbb{Z}\mathrm{Polyhedron}(P, L)$ is **normalized** if
   a. it is non-empty, and P is given by a system of linear inequalities of the form

$$\left\{ \begin{array}{ccc} a_0 & \leq x_1 \leq & b_0 \\ a_1 & \leq x_2 \leq & b_1 \\ \vdots & \vdots & \vdots \\ a_{n-1} & \leq x_d \leq & b_{n-1}, \end{array} \right. \tag{4.1}$$

   where
   b. a_i (resp. b_i) is either $-\infty$ (resp. $+\infty$) or an expression of the form $\max(\ell_{i,1} \ldots \ell_{i,e_i})$ (resp. $\min(\ell_{i,1} \ldots \ell_{i,e_i})$), and

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\text{Polyhedron}(P, L)$.
2. This is a subset of the integer points of P, which can be empty.
3. Denote by $x_1 < x_2 < \cdots < x_d$ the coordinates of $\mathbb{Z}^d$. We say that $\mathbb{Z}\text{Polyhedron}(P, L)$ is **normalized** if
   a. it is non-empty, and P is given by a system of linear inequalities of the form
   $$\begin{cases} a_0 & \leq x_1 \leq & b_0 \\ a_1 & \leq x_2 \leq & b_1 \\ \vdots & \vdots & \vdots \\ a_{n-1} & \leq x_d \leq & b_{n-1}, \end{cases} \tag{4.1}$$
   where
   b. a_i (resp. b_i) is either $-\infty$ (resp. $+\infty$) or an expression of the form $\max(\ell_{i,1} \ldots \ell_{i,e_i})$ (resp. $\min(\ell_{i,1} \ldots \ell_{i,e_i})$), and
   c. each $\ell_{i,j} \in \mathbb{Q}[x_1, \ldots, x_{i-1}]$ with degree at most 1, so that

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\text{Polyhedron}(P, L)$.
2. This is a subset of the integer points of P, which can be empty.
3. Denote by $x_1 < x_2 < \cdots < x_d$ the coordinates of $\mathbb{Z}^d$. We say that $\mathbb{Z}\text{Polyhedron}(P, L)$ is **normalized** if
   a. it is non-empty, and P is given by a system of linear inequalities of the form

$$\left\{ \begin{array}{ccc} a_0 & \leq x_1 \leq & b_0 \\ a_1 & \leq x_2 \leq & b_1 \\ \vdots & \vdots & \vdots \\ a_{n-1} & \leq x_d \leq & b_{n-1}, \end{array} \right. \tag{4.1}$$

   where
   b. a_i (resp. b_i) is either $-\infty$ (resp. $+\infty$) or an expression of the form $\max(\ell_{i,1} \ldots \ell_{i,e_i})$ (resp. $\min(\ell_{i,1} \ldots \ell_{i,e_i})$), and
   c. each $\ell_{i,j} \in \mathbb{Q}[x_1, \ldots, x_{i-1}]$ with degree at most 1, so that
   d. all the integer points of P are obtained by back substitution, that is, by specializing x_1 to every integer value v_1 in the interval (a_0, b_0), then by specializing x_2 to every integer value v_2 in the interval $(a_1(v_1), b_1(v_1))$, and so on.

$\mathbb{Z}$-polyhedron

1. A **$\mathbb{Z}$-polyhedron** of $\mathbb{Z}^d$ is the intersection (in $\mathbb{Z}^d$) of a polyhedron $P \subseteq \mathbb{Q}^d$ and a lattice $L \subseteq \mathbb{Z}^d$; we denote it by $\mathbb{Z}\text{Polyhedron}(P, L)$.
2. This is a subset of the integer points of P, which can be empty.
3. Denote by $x_1 < x_2 < \cdots < x_d$ the coordinates of $\mathbb{Z}^d$. We say that $\mathbb{Z}\text{Polyhedron}(P, L)$ is **normalized** if
   a. it is non-empty, and P is given by a system of linear inequalities of the form
   $$\left\{ \begin{array}{ccc} a_0 & \leq x_1 \leq & b_0 \\ a_1 & \leq x_2 \leq & b_1 \\ \vdots & \vdots & \vdots \\ a_{n-1} & \leq x_d \leq & b_{n-1}, \end{array} \right. \tag{4.1}$$
   where
   b. a_i (resp. b_i) is either $-\infty$ (resp. $+\infty$) or an expression of the form $\max(\ell_{i,1} \ldots \ell_{i,e_i})$ (resp. $\min(\ell_{i,1} \ldots \ell_{i,e_i})$), and
   c. each $\ell_{i,j} \in \mathbb{Q}[x_1, \ldots, x_{i-1}]$ with degree at most 1, so that
   d. all the integer points of P are obtained by back substitution, that is, by specializing x_1 to every integer value v_1 in the interval (a_0, b_0), then by specializing x_2 to every integer value v_2 in the interval $(a_1(v_1), b_1(v_1))$, and so on.
4. The algorithm IntegerPointDecomposition [11] decomposes any $\mathbb{Z}$-polyhedron into **normalized** $\mathbb{Z}$-polyhedra.

Plan

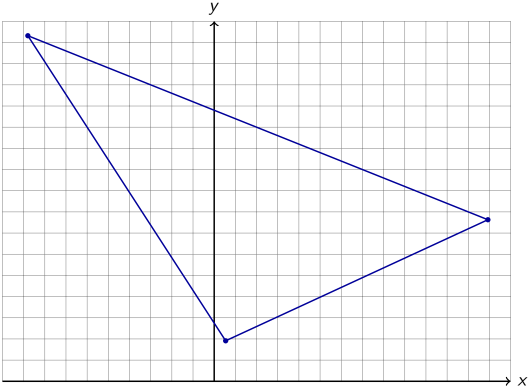
Example (0/3)

Input

Let us look at a simple example first.

Vertices: $(-44/5, 408/25), (349/27, 206/27), (85/57, 109/57)$

$$\begin{cases} 2x + 5y & \leq & 64 \\ 7x + 5y & \geq & 20 \\ 3x - 6y & \leq & -7 \end{cases}$$



Example (1/3)

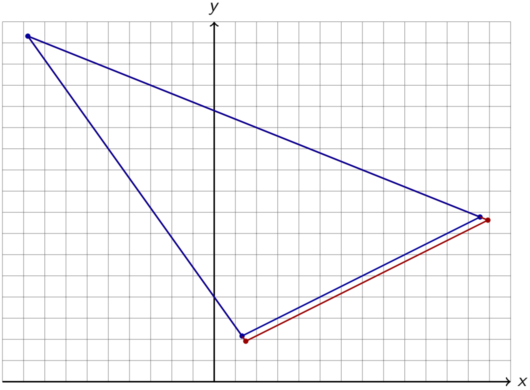
Normalization

Replace the facets that could not have integer point

Vertices: $(-44/5, 408/25)$, ~~$(349/27, 206/27)$, $(85/57, 109/57)$,~~ $(113/9, 70/9)$, $(25/19, 41/19)$

~~$3x - 6y \leq -7$~~

$$\begin{cases} 2x + 5y & \leq & 64 \\ 7x + 5y & \geq & 20 \\ 3x - 6y & \leq & -9 \end{cases}$$

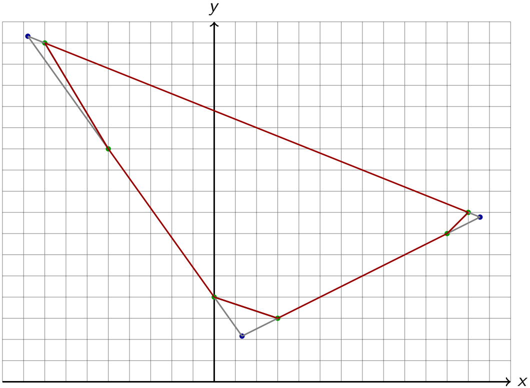


Example (2/3)

Partition

Vertices: $(-44/5, 408/25), (113/9, 70/9), (25/19, 41/19)$
Find the triangles with vertices: $[(-8, 16), (-44/5, 408/25), (-5, 11)]$, $[(3, 3), (25/19, 41/19), (0, 4)]$, $[(12, 8), (113/9, 70/9), (11, 7)]$

$$\begin{cases} 5y & \leq & -2x + 64 \\ 5y & \geq & -7x + 20 \\ 2y & \geq & x + 3 \end{cases}$$



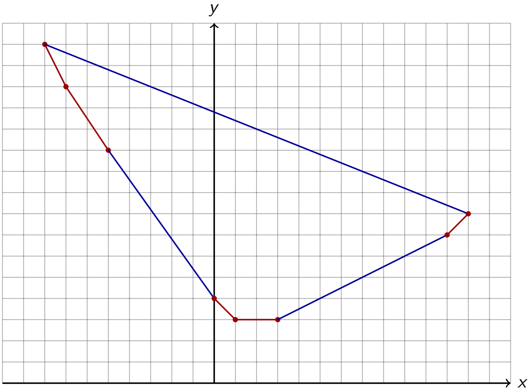
Example (3/3)

Merging

Vertices: $(-8, 16), (-7, 14), (-5, 11), (0, 4), (1, 3), (3, 3), (11, 7), (12, 8)$

$$\begin{cases} 5y & \leq & -2x + 64 \\ 5y & \geq & -7x + 20 \\ 2y & \geq & x + 3 \end{cases}$$

$$\begin{cases} y & \geq & -2x \\ 2y & \geq & 3x + 7 \\ y & \geq & -x + 4 \\ y & \geq & 3 \\ y & \geq & x - 4 \end{cases}$$



Main steps of our algorithm

Our algorithm has 3 main steps:

- **Normalization**: construct a new polyhedral set Q from P as follows. Consider in turn each facet F of P:
  1. if the hyperplane H supporting F contains an integer point, then H is a hyperplane supporting a facet of Q,
  2. otherwise we slide H towards the center of P along the normal vector of F, stopping as soon as we hit a hyperplane H' containing an integer point, then making H' a hyperplane supporting a facet of Q.

  Clearly $Q_I = P_I$.

Main steps of our algorithm

Our algorithm has 3 main steps:

- **Normalization**: construct a new polyhedral set Q from P as follows. Consider in turn each facet F of P:
  1. if the hyperplane H supporting F contains an integer point, then H is a hyperplane supporting a facet of Q,
  2. otherwise we slide H towards the center of P along the normal vector of F, stopping as soon as we hit a hyperplane H' containing an integer point, then making H' a hyperplane supporting a facet of Q.

  Clearly $Q_I = P_I$.
- **Partitioning**: make each part of the partition a polyhedron R which:
  1. either has integer points as vertices so that $R_I = R$,
  2. or has a small volume so that any algorithm (including exhaustive search) can be applied to compute R_I.

Main steps of our algorithm

Our algorithm has 3 main steps:

- **Normalization**: construct a new polyhedral set Q from P as follows. Consider in turn each facet F of P:
  1. if the hyperplane H supporting F contains an integer point, then H is a hyperplane supporting a facet of Q,
  2. otherwise we slide H towards the center of P along the normal vector of F, stopping as soon as we hit a hyperplane H' containing an integer point, then making H' a hyperplane supporting a facet of Q.

  Clearly $Q_I = P_I$.
- **Partitioning**: make each part of the partition a polyhedron R which:
  1. either has integer points as vertices so that $R_I = R$,
  2. or has a small volume so that any algorithm (including exhaustive search) can be applied to compute R_I.
- **Merging**: Once the integer hull of each part of the partition is computed and given by the list of its vertices, an algorithm for computing the convex hull of a set points, such as `QuickHull`, can be applied to deduce P_I.

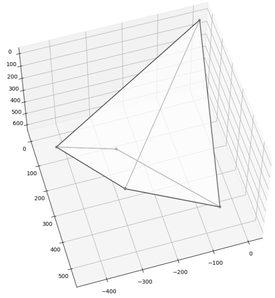
The general algorithm on a 3D example

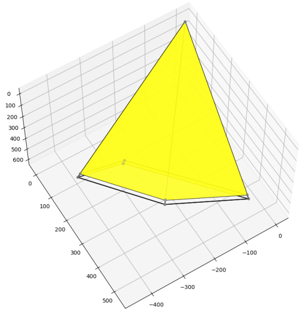
Normalization

The integer hull of the normalized polyhedral set should be the same as that of the input

$$\left\{\begin{array}{rcl} -98877x_1 - 189663x_2 - 1798x_3 & \leq & 705915 \\ -10109x_1 - 5958x_2 - 14601x_3 & \leq & 31333 \\ -5405x_1 + 4965x_2 + 3870x_3 & \leq & 4303504 \\ 729x_1 - 117x_2 + 350x_3 & \leq & 4561 \\ 677x_1 + 465x_2 - 540x_3 & \leq & 3489 \end{array}\right.$$

$$\left\{\begin{array}{rcl} -98877x_1 - 189663x_2 - 1798x_3 & \leq & 705915 \\ -10109x_1 - 5958x_2 - 14601x_3 & \leq & 31333 \\ -1081x_1 + 993x_2 + 774x_3 & \leq & 860700 \\ 729x_1 - 117x_2 + 350x_3 & \leq & 4561 \\ 677x_1 + 465x_2 - 540x_3 & \leq & 3489 \end{array}\right.$$





skip slide

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:
   a. v,

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:
   a. v,
   b. the "closest integer point" to v on f,

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:
   a. v,
   b. the "closest integer point" to v on f,
   c. all the "closest integer point" to v on F, for $F \in \mathcal{F}$.

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:
   a. v,
   b. the "closest integer point" to v on f,
   c. all the "closest integer point" to v on F, for $F \in \mathcal{F}$.
3. if there is no integer point on f, a single "corner" polyhedral set is built for f as the convex hull of:

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:
   - a. v,
   - b. the "closest integer point" to v on f,
   - c. all the "closest integer point" to v on F, for $F \in \mathcal{F}$.
3. if there is no integer point on f, a single "corner" polyhedral set is built for f as the convex hull of:
   - a. the vertex set of f,

The general algorithm: building the partition

Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:
   a. v,
   b. the "closest integer point" to v on f,
   c. all the "closest integer point" to v on F, for $F \in \mathcal{F}$.
3. if there is no integer point on f, a single "corner" polyhedral set is built for f as the convex hull of:
   a. the vertex set of f,
   b. all the closest integer point to v on F, for each vertex v of f, for each $F \in \mathcal{F}$.

The general algorithm: building the partition

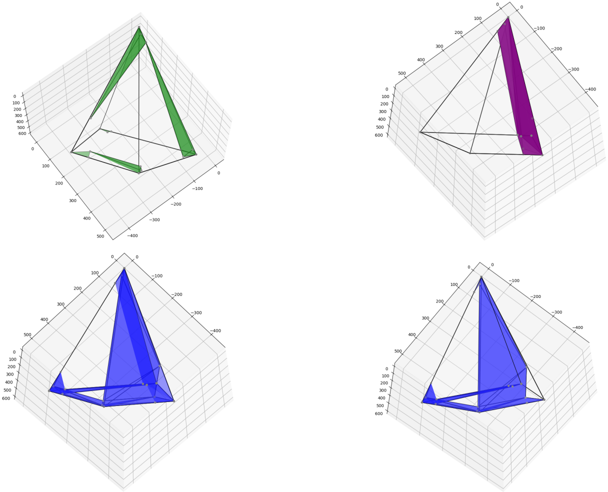
Partition

For each face f (of positive dimension) of P:

1. let $\mathcal{F}$ be the set of all facets that intersect at f
2. if there exist integer points on f (which implies that the closest integer points on f to each of its vertices do exist as well), then for each vertex v of f, a "corner" polyhedral is built as the convex hull of:
   a. v,
   b. the "closest integer point" to v on f,
   c. all the "closest integer point" to v on F, for $F \in \mathcal{F}$.
3. if there is no integer point on f, a single "corner" polyhedral set is built for f as the convex hull of:
   a. the vertex set of f,
   b. all the closest integer point to v on F, for each vertex v of f, for each $F \in \mathcal{F}$.
4. See [15] and the PhD thesis of Lin-Xiao Wang.

The general algorithm on a 3D example

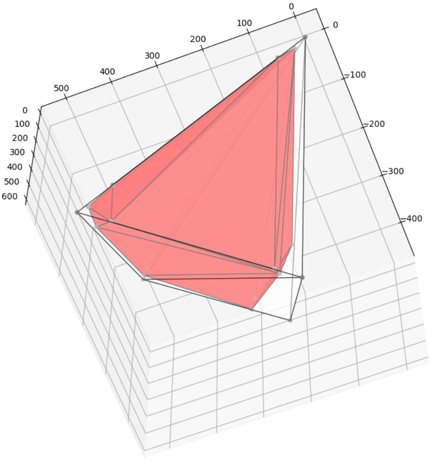
Partition

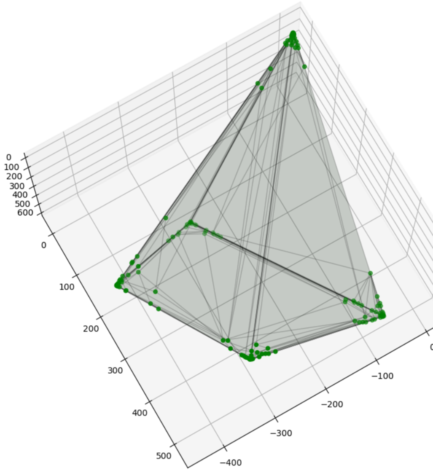


The general algorithm on a 3D example

Merging

The integer hull has 139 vertices





“Closest integer points” on a facet to each of its vertices

Projection and recursive call

In $\mathbb{Q}^d$, for a facet F of dimension $d-1$, and its vertex set V:

“Closest integer points” on a facet to each of its vertices

Projection and recursive call

In $\mathbb{Q}^d$, for a facet F of dimension $d-1$, and its vertex set V:

1. make a projection on a full-dimensional polyhedron G using Hermite normal form $\vec{c}^t U = [\mathbf{0} H]$ (where $U = [U_L U_R]$ and $\vec{c}^t \mathbf{x} = s$ is the hyperplane supporting F)

“Closest integer points” on a facet to each of its vertices

Projection and recursive call

In $\mathbb{Q}^d$, for a facet F of dimension $d-1$, and its vertex set V:

1. make a projection on a full-dimensional polyhedron G using Hermite normal form $\vec{c}^t U = [\mathbf{0} H]$ (where $U = [U_L U_R]$ and $\vec{c}^t \mathbf{x} = s$ is the hyperplane supporting F)
2. we obtain a parametrization R_F of F of the form:

$$R_F : \begin{cases} \mathbb{Q}^{d-1} & \rightarrow & \mathbb{Q}^d \\ \mathbf{z} & \longmapsto & \mathbf{x} = \mathbf{v} + U_L \mathbf{z}. \end{cases} \tag{4.2}$$

“Closest integer points” on a facet to each of its vertices

Projection and recursive call

In $\mathbb{Q}^d$, for a facet F of dimension $d-1$, and its vertex set V:

1. make a projection on a full-dimensional polyhedron G using Hermite normal form $\vec{c}^{\,t}U = [\mathbf{0}H]$ (where $U = [U_L U_R]$ and $\vec{c}^{\,t}\mathbf{x} = s$ is the hyperplane supporting F)
2. we obtain a parametrization R_F of F of the form:

$$R_F : \begin{cases} \mathbb{Q}^{d-1} & \rightarrow & \mathbb{Q}^d \\ \mathbf{z} & \longmapsto & \mathbf{x} = \mathbf{v} + U_L\mathbf{z}. \end{cases} \tag{4.2}$$

3. thus $R_F(G) = F$. Moreover, we have

$$R_F(G_I) = F_I.$$

“Closest integer points” on a facet to each of its vertices

Projection and recursive call

In $\mathbb{Q}^d$, for a facet F of dimension $d-1$, and its vertex set V:

1. make a projection on a full-dimensional polyhedron G using Hermite normal form $\vec{c}^t U = [\mathbf{0} H]$ (where $U = [U_L U_R]$ and $\vec{c}^t \mathbf{x} = s$ is the hyperplane supporting F)
2. we obtain a parametrization R_F of F of the form:
$$R_F : \begin{cases} \mathbb{Q}^{d-1} & \to & \mathbb{Q}^d \\ \mathbf{z} & \longmapsto & \mathbf{x} = \mathbf{v} + U_L \mathbf{z}. \end{cases} \tag{4.2}$$
3. thus $R_F(G) = F$. Moreover, we have
$$R_F(G_I) = F_I.$$
4. q recursive call to our integer hull algorithm computes the vertices V_I' of the integer hull of G

“Closest integer points” on a facet to each of its vertices

Projection and recursive call

In $\mathbb{Q}^d$, for a facet F of dimension $d-1$, and its vertex set V:

1. make a projection on a full-dimensional polyhedron G using Hermite normal form $\vec{c}^t U = [\mathbf{0} H]$ (where $U = [U_L U_R]$ and $\vec{c}^t \mathbf{x} = s$ is the hyperplane supporting F)
2. we obtain a parametrization R_F of F of the form:
$$R_F : \begin{cases} \mathbb{Q}^{d-1} & \rightarrow & \mathbb{Q}^d \\ \mathbf{z} & \longmapsto & \mathbf{x} = \mathbf{v} + U_L \mathbf{z}. \end{cases} \tag{4.2}$$
3. thus $R_F(G) = F$. Moreover, we have
$$R_F(G_I) = F_I.$$
4. q recursive call to our integer hull algorithm computes the vertices V_I' of the integer hull of G
5. we deduce the vertices V_I of F_I by $R_F(V_I') = V_I$

“Closest integer points” on a facet to each of its vertices

Projection and recursive call

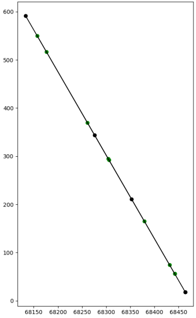
In $\mathbb{Q}^d$, for a facet F of dimension $d-1$, and its vertex set V:

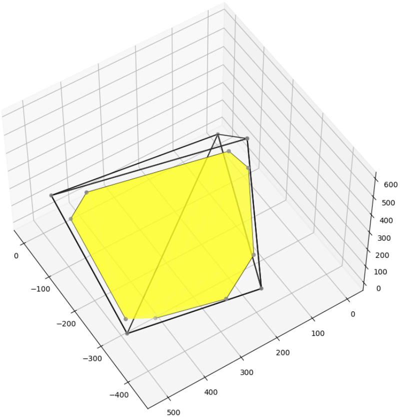
1. make a projection on a full-dimensional polyhedron G using Hermite normal form $\vec{c}^t U = [\mathbf{0} H]$ (where $U = [U_L U_R]$ and $\vec{c}^t \mathbf{x} = s$ is the hyperplane supporting F)
2. we obtain a parametrization R_F of F of the form:
$$R_F : \begin{cases} \mathbb{Q}^{d-1} & \rightarrow & \mathbb{Q}^d \\ \mathbf{z} & \longmapsto & \mathbf{x} = \mathbf{v} + U_L \mathbf{z}. \end{cases} \tag{4.2}$$
3. thus $R_F(G) = F$. Moreover, we have
$$R_F(G_I) = F_I.$$
4. q recursive call to our integer hull algorithm computes the vertices V_I' of the integer hull of G
5. we deduce the vertices V_I of F_I by $R_F(V_I') = V_I$
6. finally, we find in V_I the “closest integer points” to each v of V.

Closest integer points on a face to one of its vertices

Projection and recursive call

$$R_F : \begin{cases} x_1 = 993x_1' + 573x_2' - 67995300 \\ x_2 = 1081x_1' + 623x_2' - 74020200 \\ x_3 = x_2' \end{cases}$$





The `PolyhedralSets:-IntegerHull` command in MAPLE

```
> with(PolyhedralSets) :
> ineqs := [2 x + 5 y ≤ 64, 7 x + 5 y ≥ 20 , 3 x − 6 y ≤ -7] :
> poly := PolyhedralSet(ineqs, [x, y]);
```

$$poly := \begin{cases} Coordinates & : & [x, y] \\ Relations & : & \left[-x - \frac{5\,y}{7} \leq -\frac{20}{7},\, x - 2\,y \leq -\frac{7}{3},\, x + \frac{5\,y}{2} \leq 32 \right] \end{cases}$$

```
> IntegerHull(poly);
```

$$[[[12, 8], [-8, 16], [-7, 14], [-5, 11], [0, 4], [1, 3], [3, 3], [11, 7]], [\ ]]$$

```
> IntegerHull(poly, returntype=polyhedralset);
```

$$\begin{cases} Coordinates & : & [x, y] \\ Relations & : & \left[-y \leq -3,\ -x - y \leq -4,\ -x - \frac{5\,y}{7} \leq -\frac{20}{7},\ -x - \frac{2\,y}{3} \leq -\frac{7}{3},\ -x - \frac{y}{2} \leq 0,\, x - 2\,y \leq - \right. \end{cases}$$

```
>
```

The PolyhedralSets:-IntegerHull command in Maple

```
> restart; with(PolyhedralSets) :
> vertices := [[10, 10, 10, 10/3], [-140/8, -220/12, -10, -10/3], [60/8, 20, -100/12, -70/3], [
      /4, -100/12, 70/2, 35/3], [0, 0, 0, 50/3]] :
  vars := [x1, x2, x3, x4] :
  poly := PolyhedralSet(vertices, [ ], vars);
```

$$poly := \begin{cases} Coordinates : [x1, x2, x3, x4] \\ Relations : \left[-x1 + \frac{503\, x2}{694} + \frac{85\, x3}{694} + \frac{311\, x4}{2082} \leq \frac{7775}{3123}, -x1 + \frac{2715\, x2}{2234} + \frac{603\, x3}{2234} \right. \end{cases}$$

```
> IntegerHull(poly);
```

[[[−15, −16, −6, −2], [−15, −15, −9, −4], [−14, −15, −4, −1], [−13, −13, −8, −1],
[−13, −12, −9, −5], [−12, −13, −4, 1], [−12, −13, −4, 2], [−12, −12, −3, −3], [−1
−12, −3, −1], [−11, −11, −6, −3], [−11, −11, −1, −3], [−10, −8, −8, −5], [−9, −
−8, −8], [−7, −7, −4, 7], [−7, −6, −5, 3], [−7, −3, −8, −10], [−7, −3, −7, −10],
[−6, −4, −5, 0], [−5, −9, 23, 8], [−5, −4, −4, 5], [−5, −4, −3, −2], [−4, −5, 3, 10],
[−4, −5, 3, 11], [−4, −3, −2, −3], [−4, 0, −6, −9], [−3, −8, 30, 10], [−3, −7, 23, 10
[−3, −7, 23, 11], [−3, −6, 24, 6], [−3, 3, −8, −12], [−3, 3, −7, −13], [−2, −7, 31, 11
[−2, −6, 24, 8], [−2, −6, 24, 12], [−2, −6, 26, 11], [−2, −5, 25, 7], [−2, −5, 26, 6],

The PolyhedralSets:-IntegerHull command in MAPLE

```
> ineqs := [-x1-(132*x2)/205-(62*x3)/205 <= -1358/123, -x1 + (34*x2)/34 + (4*x3)/4
      <= 1405/17, x1-(12*x2)/118 + (83*x3)/177 <= 3500/59]:
  poly := PolyhedralSet(ineqs, [x1, x2, x3]);
  IsBounded(poly);
```

$$poly := \begin{cases} Coordinates : [x1, x2, x3] \\ Relations : \left[-x1 - \frac{132\,x2}{205} - \frac{62\,x3}{205} \leq -\frac{1358}{123}, -x1 + x2 + x3 \leq \frac{1405}{17}, x1 - \frac{6\,x2}{59}\right. \end{cases}$$

false

```
> IntegerHull(poly);
```

$$\Big[[[-20, 36, 26], [-4, -25, 103], [-2, -30, 107], [-1, -36, 117], [0, -38, 118], [0, -36, 118], [1, -37, 112], [1, -34, 117], [2, -39, 113], [2, -38, 114], [10, -43, 95], [26, -51, 60], [399, -238, -776], [403, -240, -785], [453, -265, -897], [1544, -811, -3342]], \left[\left[\frac{101}{260}, 1, -\frac{159}{260}\right], \left[\frac{2012}{4509}, -\frac{6041}{27054}, -1\right], \left[-\frac{70}{337}, \frac{267}{337}, -1\right]\right]\Big]$$

Benchmarks 2D

E&C represents "enumeration and convex hull", which in Maple is done by `ZPolyhedralSets:-EnumerateIntegerPoints` and `ConvexHull`. `Normaliz` is an open source tool for computations in affine monoids, vector configurations, lattice polytopes, and rational cones.

Volume	27.95	null	111.79	null	11179.32	null
Algorithm	IntegerHull	E&C	IntegerHull	E&C	IntegerHull	E&C
Maple (ms)	172	410	244	890	159	58083
C/C++ (ms)	0.284	0.768	0.339	1.676	0.286	6.883
Normaliz (ms)	835.730	null	462.116	null	1559.401	null

Table: Integer hulls of triangles

Volume	58.21	null	5820.95	null	23283.82	null
Algorithm	IntegerHull	E&C	IntegerHull	E&C	IntegerHull	E&C
Maple (ms)	303	752	275	31357	304	123159
C/C++ (ms)	0.451	0.565	0.478	0.657	0.396	0.682
Normaliz (ms)	2.837	null	1216.238	null	740.559	null

Table: Integer hulls of hexagons

Benchmarks 3D

Volume	447.48	null	6991.89	null	55935.2	null
Algorithm	IntegerHull	E&C	IntegerHull	E&C	IntegerHull	E&C
Maple (ms)	977	7289	1223	74804	1378	531904
C/C++ (ms)	4.488	0.826	4.615	0.923	4.624	1.527
Normaliz (ms)	851.495	null	956.666	null	793.192	null

Table: Integer hulls of tetrahedrons (4 vertices, 4 facets and 6 edges)

Volume	412.58	null	7050.81	null	60417.63	null
Algorithm	IntegerHull	E&C	IntegerHull	E&C	IntegerHull	E&C
Maple (ms)	1476	5711	1573	60233	1728	512101
C/C++ (ms)	11.049	21.235	16.001	145.068	23.822	2082.559
Normaliz (ms)	7862.109	null	N/A	null	N/A	null

Table: Integer hulls of triangular bipyramids (5 vertices, 6 facets and 9 edges)

Conclusions

1. We implemented the proposed algorithm for in both MAPLE and C/C++.
2. Our algorithm takes into consideration the geometric properties of the input polyhedral set.
3. That is, if the input polyhedral set is close to be its own integer hull, then computations are cheaper
4. Moreover, the cost of the computations depend mainly on the shape of the input while the size of the input has little impact.
5. Doing a complexity analysis that would reflect that fact is an open problem to us.

Plan

Plan

Counting memory accesses in a for-loop nest (1/2)

Consider the well-known example SOR (Successive-Over Relaxation) from the numerical solving of PDEs (Partial differential Equations).

```
for (i=2, i<N, i++) null
    for (j=2, j <N, j++) null
        a[i][j] = (2*a[i][j] + a[i-1][j] + a[i+1][j] + null
                   a[i][j-1] + a[i][j+1])/6; null
```

Counting memory accesses in a for-loop nest (1/2)

Consider the well-known example SOR (Successive-Over Relaxation) from the numerical solving of PDEs (Partial differential Equations).

```
for (i=2, i<N, i++) null
    for (j=2, j <N, j++) null
        a[i][j] = (2*a[i][j] + a[i-1][j] + a[i+1][j] + null
                   a[i][j-1] + a[i][j+1])/6; null
```

- The memory slots accessed by the for-loop nest are given by:

$$\{(i + \Delta i, j + \Delta j) \quad | \quad -1 \le \Delta i - \Delta j, \Delta i + \Delta j, \le 1, 2 \le i, j \le N - 1\}$$

Counting memory accesses in a for-loop nest (1/2)

Consider the well-known example SOR (Successive-Over Relaxation) from the numerical solving of PDEs (Partial differential Equations).

```
for (i=2, i<N, i++) null
    for (j=2, j <N, j++) null
        a[i][j] = (2*a[i][j] + a[i-1][j] + a[i+1][j] + null
                   a[i][j-1] + a[i][j+1])/6; null
```

- The memory slots accessed by the for-loop nest are given by:

  $$\{(i + \Delta i, j + \Delta j) \quad | \quad -1 \leq \Delta i - \Delta j, \Delta i + \Delta j, \leq 1, 2 \leq i, j \leq N - 1\}$$

- Using standard techniques from Linear Algebra, namely Fourier-Motzkin elimination (FME), we can rewrite the above set as:

  $$\left\{ (x, y) \quad | \quad \left\{ \begin{array}{l} 1 \leq x, y \leq N \\ 3 \leq x + y \leq 2N - 1 \\ 2 - N \leq x - y \leq N - 2 \end{array} \right. \right\}, \text{ for } N \geq 3.$$

Counting memory accesses in a for-loop nest (1/2)

Consider the well-known example SOR (Successive-Over Relaxation) from the numerical solving of PDEs (Partial differential Equations).

```
for (i=2, i<N, i++) null
    for (j=2, j <N, j++) null
        a[i][j] = (2*a[i][j] + a[i-1][j] + a[i+1][j] + null
                   a[i][j-1] + a[i][j+1])/6; null
```

- The memory slots accessed by the for-loop nest are given by:

$$\{(i + \Delta i, j + \Delta j) \quad | \quad -1 \leq \Delta i - \Delta j, \Delta i + \Delta j, \leq 1, 2 \leq i, j \leq N - 1\}$$

- Using standard techniques from Linear Algebra, namely Fourier-Motzkin elimination (FME), we can rewrite the above set as:

$$\left\{ (x, y) \quad | \quad \left\{ \begin{array}{l} 1 \leq x, y \leq N \\ 3 \leq x + y \leq 2N - 1 \\ 2 - N \leq x - y \leq N - 2 \end{array} \right. \right\}, \text{ for } N \geq 3.$$

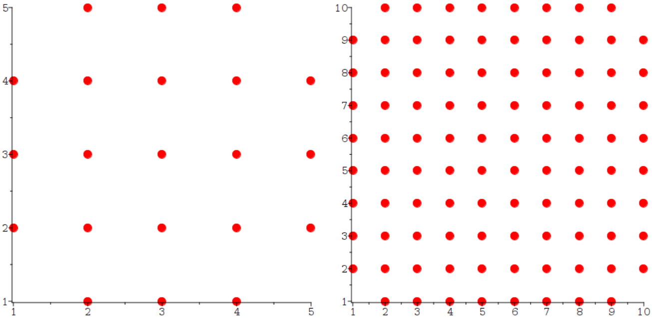
- Hence the problem becomes counting the number of integer points of a parametric polyhedral set P_N.

Counting memory accesses in a for-loop nest (2/2)

```
for (i=2, i<N, i++) null
    for (j=2, j <N, j++) null
        a[i][j] = (2*a[i][j] + a[i-1][j] + a[i+1][j] + null
                   a[i][j-1] + a[i][j+1])/6; null
```

Counting memory accesses in a for-loop nest (2/2)

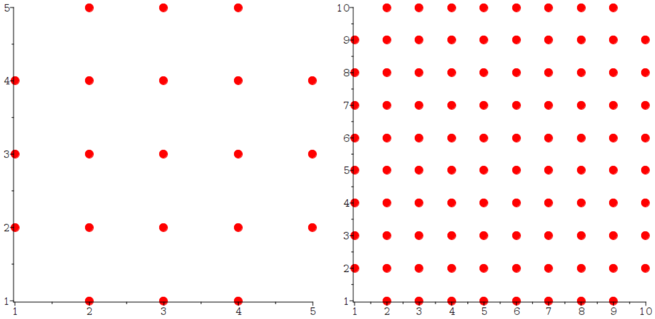
```
for (i=2, i<N, i++)
    for (j=2, j <N, j++)
        a[i][j] = (2*a[i][j] + a[i-1][j] + a[i+1][j] +
                   a[i][j-1] + a[i][j+1])/6;
```



The integer points of the parametric polyhedron P_N for $N = 5$ and $N = 10$.

Counting memory accesses in a for-loop nest (2/2)

```
for (i=2, i<N, i++) null
    for (j=2, j <N, j++) null
        a[i][j] = (2*a[i][j] + a[i-1][j] + a[i+1][j] + null
                   a[i][j-1] + a[i][j+1])/6; null
```



The integer points of the parametric polyhedron P_N for $N = 5$ and $N = 10$.
We will see later that $\mid P \cap \mathbb{Z}^2 \mid = N^2 - 4$.

Objective and challenges

1. Given a parametric polyhedron $P(\vec{b})$, we want to count the number of its integer points as a function $c(\vec{b})$ of the parameter $\vec{b}$.

Objective and challenges

1. Given a parametric polyhedron $P(\vec{b})$, we want to count the number of its integer points as a function $c(\vec{b})$ of the parameter $\vec{b}$.
2. One challenge is that the shape (vertices, facets, etc.) of the integer hull of $P(\vec{b})$, that is, $P(\vec{b}) \cap \mathbb{Z}^d$, may vary with the values of $\vec{b}$.

Objective and challenges

1. Given a parametric polyhedron $P(\vec{b})$, we want to count the number of its integer points as a function $c(\vec{b})$ of the parameter $\vec{b}$.
2. One challenge is that the shape (vertices, facets, etc.) of the integer hull of $P(\vec{b})$, that is, $P(\vec{b}) \cap \mathbb{Z}^d$, may vary with the values of $\vec{b}$.
3. Consider the parametric polyhedron P_N given by:

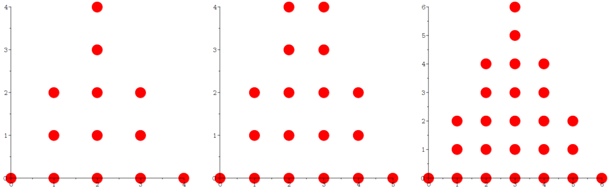
$$\begin{cases} 0 \leq i, j \\ j \leq 2i \\ 2i + j \leq N \end{cases}$$

Objective and challenges

1. Given a parametric polyhedron $P(\vec{b})$, we want to count the number of its integer points as a function $c(\vec{b})$ of the parameter $\vec{b}$.
2. One challenge is that the shape (vertices, facets, etc.) of the integer hull of $P(\vec{b})$, that is, $P(\vec{b}) \cap \mathbb{Z}^d$, may vary with the values of $\vec{b}$.
3. Consider the parametric polyhedron P_N given by:

$$\begin{cases} 0 \leq i, j \\ j \leq 2i \\ 2i + j \leq N \end{cases}$$

4. The plots below show P_N for $N = 8, 10, 12$.

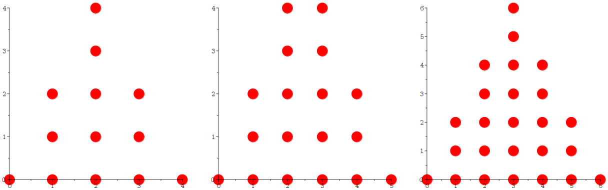


Objective and challenges

1. Given a parametric polyhedron $P(\vec{b})$, we want to count the number of its integer points as a function $c(\vec{b})$ of the parameter $\vec{b}$.
2. One challenge is that the shape (vertices, facets, etc.) of the integer hull of $P(\vec{b})$, that is, $P(\vec{b}) \cap \mathbb{Z}^d$, may vary with the values of $\vec{b}$.
3. Consider the parametric polyhedron P_N given by:

$$\begin{cases} 0 \leq i, j \\ j \leq 2i \\ 2i + j \leq N \end{cases}$$

4. The plots below show P_N for $N = 8, 10, 12$.



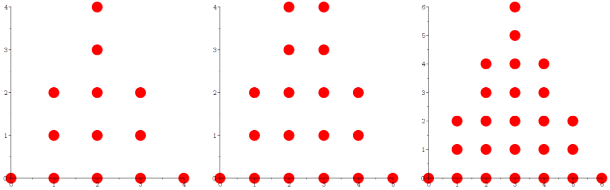
5. Fortunately, Ehrhart Theory tells us that these variations are periodic

Objective and challenges

1. Given a parametric polyhedron $P(\vec{b})$, we want to count the number of its integer points as a function $c(\vec{b})$ of the parameter $\vec{b}$.
2. One challenge is that the shape (vertices, facets, etc.) of the integer hull of $P(\vec{b})$, that is, $P(\vec{b}) \cap \mathbb{Z}^d$, may vary with the values of $\vec{b}$.
3. Consider the parametric polyhedron P_N given by:

$$\begin{cases} 0 \leq i, j \\ j \leq 2i \\ 2i + j \leq N \end{cases}$$

4. The plots below show P_N for $N = 8, 10, 12$.



5. Fortunately, Ehrhart Theory tells us that these variations are periodic
6. Hence, the function $c(\vec{b})$ is computable as a piece-wise function.

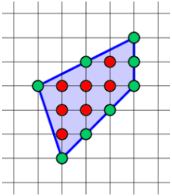
Related works

- Given a 2D polytope P, whose vertices are integer points, Pick's theorem relates the area A of P, the number b of integer points on the border of P, and the number i in the interior of P:

$$A = i + \frac{b}{2} - 1$$

Related works

- Given a 2D polytope P, whose vertices are integer points, Pick's theorem relates the area A of P, the number b of integer points on the border of P, and the number i in the interior of P:
$$A = i + \frac{b}{2} - 1$$

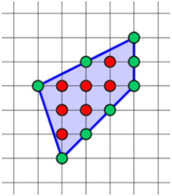


Related works

- Given a 2D polytope P, whose vertices are integer points, Pick's theorem relates the area A of P, the number b of integer points on the border of P, and the number i in the interior of P:

$$A = i + \frac{b}{2} - 1$$

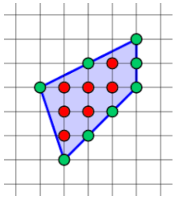
- No generalization of Pick's theorem to higher dimension.



Related works

- Given a 2D polytope P, whose vertices are integer points, Pick's theorem relates the area A of P, the number b of integer points on the border of P, and the number i in the interior of P:

$$A = i + \frac{b}{2} - 1$$

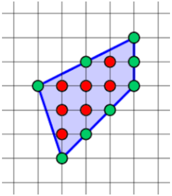


- No generalization of Pick's theorem to higher dimension.
- By studying the dilation of polyhedral sets, Eugène Ehrhart discovered and studied the **periodic behaviour** of parametric polyhedral sets.

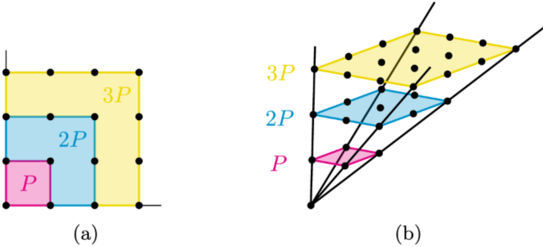
Related works

- Given a 2D polytope P, whose vertices are integer points, Pick's theorem relates the area A of P, the number b of integer points on the border of P, and the number i in the interior of P:

$$A = i + \frac{b}{2} - 1$$

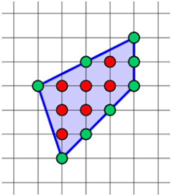


- No generalization of Pick's theorem to higher dimension.
- By studying the dilation of polyhedral sets, Eugène Ehrhart discovered and studied the **periodic behaviour** of parametric polyhedral sets.

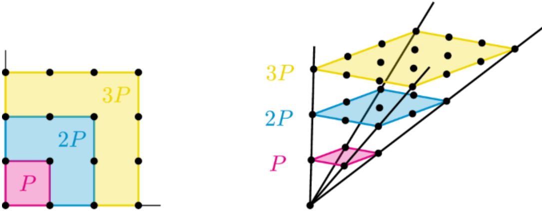


Related works

- Given a 2D polytope P, whose vertices are integer points, Pick's theorem relates the area A of P, the number b of integer points on the border of P, and the number i in the interior of P:
$$A = i + \frac{b}{2} - 1$$
- No generalization of Pick's theorem to higher dimension.



- By studying the dilation of polyhedral sets, Eugène Ehrhart discovered and studied the **periodic behaviour** of parametric polyhedral sets.
- See Ehrhart polynomial.
- Images are from Wikipedia (fair use category).



(a) (b)

Plan

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^{\mathbf{e}}.$$

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^{\mathbf{e}}.$$

- If P is bounded, then $G(P, (1, \ldots, 1))$ counts the number of its integer points.

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^{\mathbf{e}}.$$

- If P is bounded, then $G(P, (1, \ldots, 1))$ counts the number of its integer points.
- If P is not bounded, then $G(P, \mathbf{x})$ is a formal power series and can still be manipulated algorithmically.

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^{\mathbf{e}}.$$

- If P is bounded, then $G(P, (1, \ldots, 1))$ counts the number of its integer points.
- If P is not bounded, then $G(P, \mathbf{x})$ is a formal power series and can still be manipulated algorithmically.
- For $d = 2$, suppose P is the ray corresponding to $y = 0$ and $x \geq 0$, then:

$$G(P, \mathbf{x}) =$$

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^\mathbf{e} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^\mathbf{e}.$$

- If P is bounded, then $G(P, (1, \ldots, 1))$ counts the number of its integer points.
- If P is not bounded, then $G(P, \mathbf{x})$ is a formal power series and can still be manipulated algorithmically.
- For $d = 2$, suppose P is the ray corresponding to $y = 0$ and $x \geq 0$, then:

$$G(P, \mathbf{x}) = \sum_{n=0}^{n=\infty} (x, y)^{(n,0)} =$$

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^{\mathbf{e}}.$$

- If P is bounded, then $G(P, (1, \ldots, 1))$ counts the number of its integer points.
- If P is not bounded, then $G(P, \mathbf{x})$ is a formal power series and can still be manipulated algorithmically.
- For $d = 2$, suppose P is the ray corresponding to $y = 0$ and $x \geq 0$, then:

$$G(P, \mathbf{x}) = \sum_{n=0}^{n=\infty} (x, y)^{(n,0)} = \sum_{n=0}^{n=\infty} x^n y^0 =$$

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^{\mathbf{e}}.$$

- If P is bounded, then $G(P, (1, \ldots, 1))$ counts the number of its integer points.
- If P is not bounded, then $G(P, \mathbf{x})$ is a formal power series and can still be manipulated algorithmically.
- For $d = 2$, suppose P is the ray corresponding to $y = 0$ and $x \geq 0$, then:

$$G(P, \mathbf{x}) = \sum_{n=0}^{n=\infty} (x, y)^{(n,0)} = \sum_{n=0}^{n=\infty} x^n y^0 = \sum_{n=0}^{n=\infty} x^n =$$

Generating function of a polyhedral set (1/4)

- Consider a polyhedral set $P \subseteq \mathbb{Q}^d$.
- Each integer point $\mathbf{e} = (e_1, \ldots, e_d)$ of P is mapped to the monomial $\mathbf{x}^{\mathbf{e}} = {x_1}^{e_1} \cdots {x_d}^{e_d}$
- When $d = 2$, we write (x, y) instead of (x_1, x_2).

Definition

The **generating function** of P is the formal Laurent series:

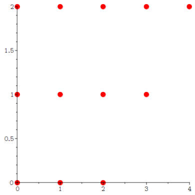
$$G(P, \mathbf{x}) = \sum_{\mathbf{e} \in P \cap \mathbb{Z}^d} \mathbf{x}^{\mathbf{e}}.$$

- If P is bounded, then $G(P, (1, \ldots, 1))$ counts the number of its integer points.
- If P is not bounded, then $G(P, \mathbf{x})$ is a formal power series and can still be manipulated algorithmically.
- For $d = 2$, suppose P is the ray corresponding to $y = 0$ and $x \geq 0$, then:

$$G(P, \mathbf{x}) = \sum_{n=0}^{n=\infty} (x, y)^{(n,0)} = \sum_{n=0}^{n=\infty} x^n y^0 = \sum_{n=0}^{n=\infty} x^n = \frac{1}{1 - x}.$$

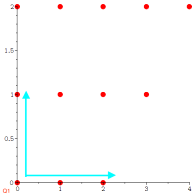
Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.



Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.

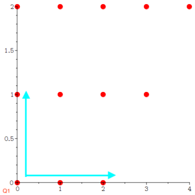


Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x,,y)$ with $x, y \geq 0$. Then, we have:

$$G(Q_1, \mathbf{x}) =$$

Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.

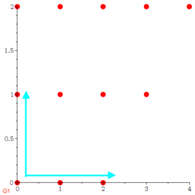


Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x, , y)$ with $x, y \geq 0$. Then, we have:

$$G(Q_1, \mathbf{x}) = \sum_{m,n \geq 0} x^m y^n =$$

Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.

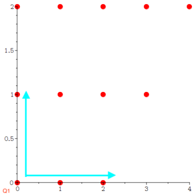


Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x, , y)$ with $x, y \geq 0$. Then, we have:

$$G(Q_1, \mathbf{x}) = \sum_{m,n \geq 0} x^m y^n = \left(\sum_{m=0}^{n=\infty} x^m \right) \left(\sum_{n=0}^{n=\infty} y^n \right) =$$

Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.

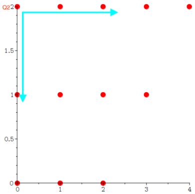


Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x, , y)$ with $x, y \geq 0$. Then, we have:

$$G(Q_1, \mathbf{x}) = \sum_{m,n \geq 0} x^m y^n = \left(\sum_{m=0}^{n=\infty} x^m \right) \left(\sum_{n=0}^{n=\infty} y^n \right) = \frac{1}{1-x} \frac{1}{1-y}.$$

Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.



Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x, , y)$ with $x, y \geq 0$. Then, we have:

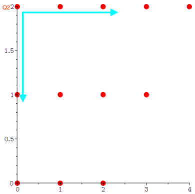
$$G(Q_1, \mathbf{x}) = \sum_{m,n \geq 0} x^m y^n = \left(\sum_{m=0}^{n=\infty} x^m \right) \left(\sum_{n=0}^{n=\infty} y^n \right) = \frac{1}{1-x} \frac{1}{1-y}.$$

Consider the top-left corner of P, that is, the vertex cone Q_2 rooted at $(0, 2)$ and with rays $(0, 1)$ and $(1, 0)$.

$G(Q_2, \mathbf{x}) =$

Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.



Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x, , y)$ with $x, y \geq 0$. Then, we have:

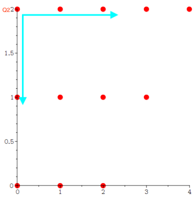
$$G(Q_1, \mathbf{x}) = \sum_{m,n\geq 0} x^m y^n = \left(\sum_{m=0}^{n=\infty} x^m\right)\left(\sum_{n=0}^{n=\infty} y^n\right) = \frac{1}{1-x}\frac{1}{1-y}.$$

Consider the top-left corner of P, that is, the vertex cone Q_2 rooted at $(0, 2)$ and with rays $(0, 1)$ and $(1, 0)$.

$$G(Q_2, \mathbf{x}) = \left(\sum_{m\geq 0} x^m\right)\left(\sum_{n\leq 2} y^n\right) =$$

Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.



Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x, , y)$ with $x, y \geq 0$. Then, we have:

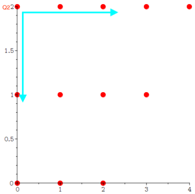
$$G(Q_1, \mathbf{x}) = \sum_{m,n \geq 0} x^m y^n = \left(\sum_{m=0}^{n=\infty} x^m \right) \left(\sum_{n=0}^{n=\infty} y^n \right) = \frac{1}{1-x} \frac{1}{1-y}.$$

Consider the top-left corner of P, that is, the vertex cone Q_2 rooted at $(0, 2)$ and with rays $(0, 1)$ and $(1, 0)$.

$$G(Q_2, \mathbf{x}) = \left(\sum_{m \geq 0} x^m \right) \left(\sum_{n \leq 2} y^n \right) = \left(\sum_{m \geq 0} x^m \right) y^2 \left(\sum_{n \geq 0} (y^{-1})^n \right) =$$

Generating function of a polyhedral set (2/4)

With $d = 2$, we will compute $G(P, \mathbf{x})$ for the polyhedron P given as the convex hull of the 12 points on the figure below.



Consider the bottom-left of P, that is, the first quadrant Q_1, that is, the points $(x, , y)$ with $x, y \geq 0$. Then, we have:

$$G(Q_1, \mathbf{x}) = \sum_{m,n\geq 0} x^m y^n = \left(\sum_{m=0}^{n=\infty} x^m\right)\left(\sum_{n=0}^{n=\infty} y^n\right) = \frac{1}{1-x}\frac{1}{1-y}.$$

Consider the top-left corner of P, that is, the vertex cone Q_2 rooted at $(0, 2)$ and with rays $(0, 1)$ and $(1, 0)$.

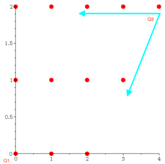
$$G(Q_2, \mathbf{x}) = \left(\sum_{m\geq 0} x^m\right)\left(\sum_{n\leq 2} y^n\right) = \left(\sum_{m\geq 0} x^m\right) y^2 \left(\sum_{n\geq 0} (y^{-1})^n\right) = \frac{1}{1-x}\frac{y^2}{1-y^{-1}}$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P

Generating function of a polyhedral set (2/4)

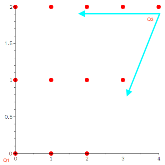
Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) =$$

Generating function of a polyhedral set (2/4)

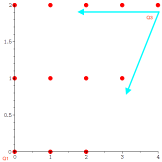
Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \le m \le 0} x^m y^n \right) =$$

Generating function of a polyhedral set (2/4)

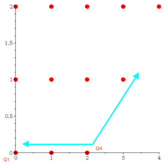
Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \leq m \leq 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P

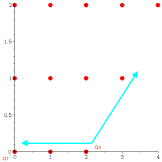


$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \le m \le 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

$$G(Q_4, \mathbf{x}) =$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P

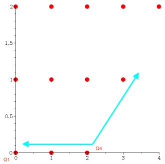


$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \le m \le 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

$$G(Q_4, \mathbf{x}) = x^4 y^0 \left(\sum_{0 \le n, m \le n} x^m y^n \right) =$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P

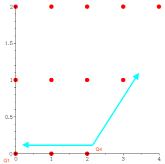


$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \le m \le 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

$$G(Q_4, \mathbf{x}) = x^4 y^0 \left(\sum_{0 \le n, m \le n} x^m y^n \right) = \frac{x^2 y^0}{(1 - xy)(1 - x^{-1})}$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \leq m \leq 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

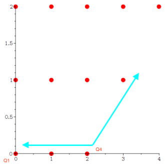
$$G(Q_4, \mathbf{x}) = x^4 y^0 \left(\sum_{0 \leq n, m \leq n} x^m y^n \right) = \frac{x^2 y^0}{(1 - xy)(1 - x^{-1})}$$

Applying a theorem of Michel Brion (1988) [3] we have:

$G(P, \mathbf{x}) \quad =$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \le m \le 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

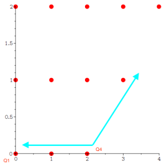
$$G(Q_4, \mathbf{x}) = x^4 y^0 \left(\sum_{0 \le n, m \le n} x^m y^n \right) = \frac{x^2 y^0}{(1 - xy)(1 - x^{-1})}$$

Applying a theorem of Michel Brion (1988) [3] we have:

$$G(P, \mathbf{x}) \quad = \quad G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x}) + G(Q_4, \mathbf{x})$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \leq m \leq 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

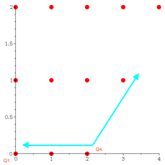
$$G(Q_4, \mathbf{x}) = x^4 y^0 \left(\sum_{0 \leq n, m \leq n} x^m y^n \right) = \frac{x^2 y^0}{(1 - xy)(1 - x^{-1})}$$

Applying a theorem of Michel Brion (1988) [3] we have:

$$\begin{aligned} G(P, \mathbf{x}) &= G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x}) + G(Q_4, \mathbf{x}) \\ &= \frac{1}{1-x} \frac{1}{1-y} + \frac{1}{1-x} \frac{y^2}{1-y^{-1}} + \frac{x^4 y^2}{(1-x^{-1})(1-x^{-1} y^{-1})} + \frac{x^2 y^0}{(1-xy)(1-x^{-1})} \end{aligned}$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \le m \le 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

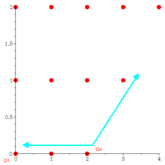
$$G(Q_4, \mathbf{x}) = x^4 y^0 \left(\sum_{0 \le n, m \le n} x^m y^n \right) = \frac{x^2 y^0}{(1 - xy)(1 - x^{-1})}$$

Applying a theorem of Michel Brion (1988) [3] we have:

$$\begin{array}{rcl} G(P, \mathbf{x}) & = & G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x}) + G(Q_4, \mathbf{x}) \\ & = & \frac{1}{1-x}\frac{1}{1-y} + \frac{1}{1-x}\frac{y^2}{1-y^{-1}} + \frac{x^4 y^2}{(1-x^{-1})(1-x^{-1}y^{-1})} + \frac{x^2 y^0}{(1-xy)(1-x^{-1})} \\ & = & y^2 + xy^2 + x^2y^2 + x^3y^2 + x^4y^2 + y + xy + x^2y + x^3y + 1 + x + x^2. \end{array}$$

Generating function of a polyhedral set (2/4)

Continuing with the other corners Q_3 and Q_4 of the polytope P



$$G(Q_3, \mathbf{x}) = x^4 y^2 \left(\sum_{n \le m \le 0} x^m y^n \right) = \frac{x^4 y^2}{(1 - x^{-1})(1 - x^{-1} y^{-1})}$$

$$G(Q_4, \mathbf{x}) = x^4 y^0 \left(\sum_{0 \le n, m \le n} x^m y^n \right) = \frac{x^2 y^0}{(1 - xy)(1 - x^{-1})}$$

Applying a theorem of Michel Brion (1988) [3] we have:

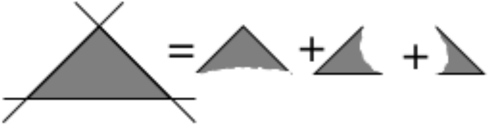
$$\begin{array}{rcl} G(P, \mathbf{x}) & = & G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x}) + G(Q_4, \mathbf{x}) \\ & = & \frac{1}{1-x} \frac{1}{1-y} + \frac{1}{1-x} \frac{y^2}{1-y^{-1}} + \frac{x^4 y^2}{(1-x^{-1})(1-x^{-1}y^{-1})} + \frac{x^2 y^0}{(1-xy)(1-x^{-1})} \\ & = & y^2 + xy^2 + x^2y^2 + x^3y^2 + x^4y^2 + y + xy + x^2y + x^3y + 1 + x + x^2. \end{array}$$

Consequently

$$\begin{array}{rcl} \mid P \cap \mathbb{Z}^2 \mid & = & G(P, (1, 1))) \\ & & = 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 \\ & & = 12. \end{array}$$

Recall Brion's formula

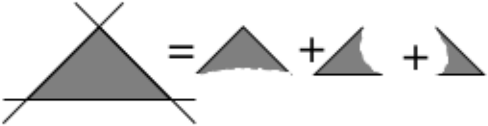
- This formula asserts that for a polytope $P \subseteq \mathbb{Q}^d$ its generating function is the sum of the generating functions of its corners (= vertex cones)



$$G(P, \mathbf{x}) = G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x})$$

Recall Brion's formula

- This formula asserts that for a polytope $P \subseteq \mathbb{Q}^d$ its generating function is the sum of the generating functions of its corners (= vertex cones)

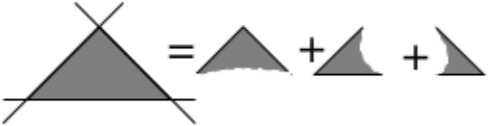


$$G(P, \mathbf{x}) \;=\; G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x})$$

- Our previous calculations used two facts

Recall Brion's formula

- This formula asserts that for a polytope $P \subseteq \mathbb{Q}^d$ its generating function is the sum of the generating functions of its corners (= vertex cones)

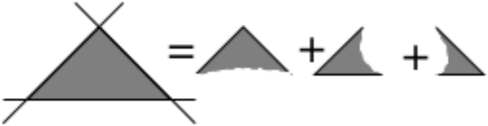


$$G(P, \mathbf{x}) \;=\; G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x})$$

- Our previous calculations used two facts
  1. In dimension $d = 2$, every cone is **simplicial** that is, can be generated by d rays,

Recall Brion's formula

- This formula asserts that for a polytope $P \subseteq \mathbb{Q}^d$ its generating function is the sum of the generating functions of its corners (= vertex cones)

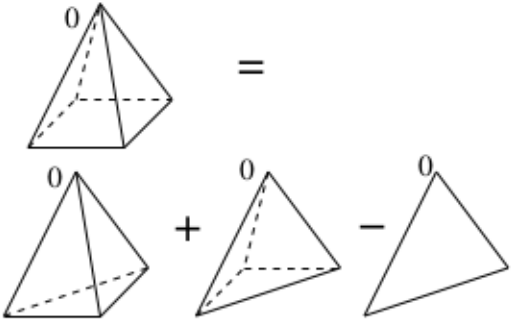


$$G(P, \mathbf{x}) = G(Q_1, \mathbf{x}) + G(Q_2, \mathbf{x}) + G(Q_3, \mathbf{x})$$

- Our previous calculations used two facts
  1. In dimension $d = 2$, every cone is **simplicial** that is, can be generated by d rays,
  2. The cones $Q2$, Q_3, Q_4 are **unimodular**, that is, the sums of the power series $G(Q_2, \mathbf{x})$, $G(Q_3, \mathbf{x})$, $G(Q_4, \mathbf{x})$ can be deduced from that of $G(Q_1, \mathbf{x})$ (the first quadrant) by means of unimodular transformations (that is, mapping integer vectors to integer vectors).

Barvinok's algorithm

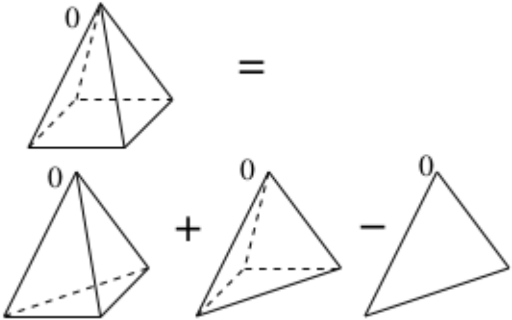
- In dimension d, one can decompose any cone into **simplicial** cones (= cones generated by d rays),



skip slide

Barvinok's algorithm

- In dimension d, one can decompose any cone into **simplicial** cones (= cones generated by d rays),

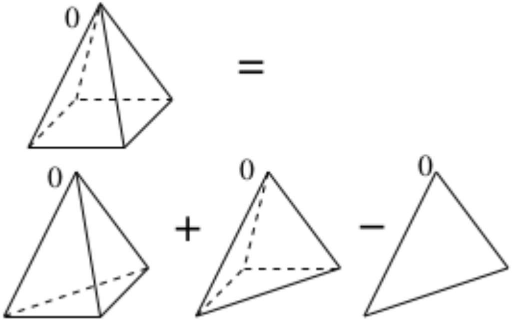


- Alexander Barvinok (1994) [2] proposed an algorithm to decompose any simplicial cones into **unimodular** cones,

⏩ skip slide

Barvinok's algorithm

- In dimension d, one can decompose any cone into **simplicial** cones ($=$ cones generated by d rays),

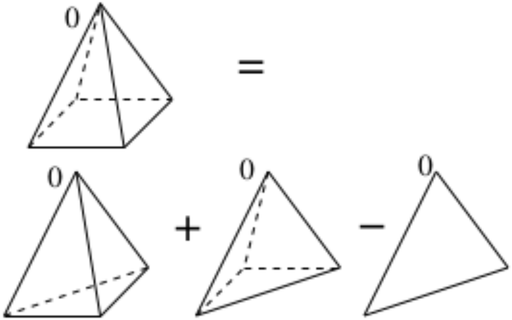


- Alexander Barvinok (1994) [2] proposed an algorithm to decompose any simplicial cones into **unimodular** cones,
- consequently, Barvinok has found the first algorithm to compute $G(P, \mathbf{x})$,

skip slide

Barvinok's algorithm

- In dimension d, one can decompose any cone into **simplicial** cones (= cones generated by d rays),



- Alexander Barvinok (1994) [2] proposed an algorithm to decompose any simplicial cones into **unimodular** cones,
- consequently, Barvinok has found the first algorithm to compute $G(P, \mathbf{x})$,
- Moreover, Barvinok's algorithm runs in **polynomial time for a fixed d.** More references on the subject: [1, 10, 14, 19, 22]

skip slide

Plan

Sanity-check examples

Example 29 (1)

Input:

$$\{1 \leq i, 1 \leq j, i \leq n, j \leq n\}$$

Output:

$$[[\{n^2\}, [0 \leq n - 1]]]$$

Sanity-check examples

Example 29 (1)

Input:

$$\{1 \leq i, 1 \leq j, i \leq n, j \leq n\}$$

Output:

$$[[\{n^2\}, [0 \leq n - 1]]]$$

Example 30 (3)

Input:

$$\{1 \leq i, 1 \leq j, i + j \leq n, 0 \leq n\}$$

Output:

$$[[\{\frac{n^2}{2} - \frac{n}{2}\}, [0 \leq n - 2]]]$$

Examples with several parameters

Example 31 (4)

Input:

$$\{1 \leq i, i \leq n, i \leq m, 1 \leq j, j \leq i\}$$

Output:

$$[[\{1\}, [m-1=0, 0 \leq n-2]],$$
$$[\{\frac{n^2}{2} + \frac{n}{2}\}, [0 \leq m-n, 0 \leq n-1]],$$
$$[\{\frac{m^2}{2} + \frac{m}{2}\}, [0 \leq m-2, 0 \leq n-3, 0 \leq -m+n-1]]]$$

Examples with several parameters

Example 31 (4)

Input:

$$\{1 \le i, i \le n, i \le m, 1 \le j, j \le i\}$$

Output:

$$\begin{gathered}[[\{1\}, [m-1=0, 0 \le n-2]],\\ [\{\tfrac{n^2}{2} + \tfrac{n}{2}\}, [0 \le m-n, 0 \le n-1]],\\ [\{\tfrac{m^2}{2} + \tfrac{m}{2}\}, [0 \le m-2, 0 \le n-3, 0 \le -m+n-1]]]\end{gathered}$$

Example 32 (5)

Input:

$$\{1 \le i, i \le n, i \le m, 1 \le j, j \le p\}$$

Output:

$$\begin{gathered}[[\{pm\}, [n-m \ge 1, p-2 \ge 0, m-1 \ge 0]],\\ [\{pn\}, [m-n \ge 0, n-2 \ge 0, p-1 \ge 0]],\\ [\{1\}, [n-1=0, p-1=0, 0 \le m-1]],\\ [\{p\}, [m-1=0, 0 \le -2+n, 0 \le p-1]]]\end{gathered}$$

Examples with quasi-polynomials

Example 33 (6)

Input:

$$\{1 \le i, j \le n, 2i \le 3j\}$$

Output:

$$[[\{Q([n, 2, [\frac{3n^2}{4} + \frac{n}{2}, -1/4 + \frac{3n^2}{4} + \frac{n}{2}]])\}, [1 \le n]]]$$

Examples with quasi-polynomials

Example 33 (6)

Input:

$$\{1 \leq i, j \leq n, 2i \leq 3j\}$$

Output:

$$[[\{Q([n, 2, [\tfrac{3n^2}{4} + \tfrac{n}{2}, -1/4 + \tfrac{3n^2}{4} + \tfrac{n}{2}]])\}, [1 \leq n]]]$$

Example 34 (7)

Input:

$$\{0 \leq i, 0 \leq j, j \leq 2i, 2i + j \leq n\}$$

Output:

$$\begin{gathered}[[\{Q([n, 4, [1 + \tfrac{n}{2} + \tfrac{n^2}{8}, 3/8 + \tfrac{n}{2} + \tfrac{n^2}{8}, 1/2 + \tfrac{n}{2} + \tfrac{n^2}{8}, 3/8 + \tfrac{n}{2} + \tfrac{n^2}{8}]])\}, \\ [0 \leq n - 1]], \\ [\{1\}, [n = 0]]]\end{gathered}$$

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones($P(\vec{b})$) determines the vertex cones (= corners) of $P(\vec{b})$

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones($P(\vec{b})$) determines the vertex cones ($=$ corners) of $P(\vec{b})$
   a. **Same challenges!**

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones($P(\vec{b})$) determines the vertex cones (= corners) of $P(\vec{b})$
   a. **Same challenges!**
   b. And, at the end, many sets of cases of the case discussion can be replaced by a single case, that is, doing **recombination**.

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones($P(\vec{b})$) determines the vertex cones (= corners) of $P(\vec{b})$
   a. **Same challenges!**
   b. And, at the end, many sets of cases of the case discussion can be replaced by a single case, that is, doing **recombination**.
3. GeneratingFunction($P(\vec{b})$) determines the generating functions of each cone Cones($P(\vec{b})$)

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones($P(\vec{b})$) determines the vertex cones (= corners) of $P(\vec{b})$
   a. **Same challenges!**
   b. And, at the end, many sets of cases of the case discussion can be replaced by a single case, that is, doing **recombination**.
3. GeneratingFunction($P(\vec{b})$) determines the generating functions of each cone Cones($P(\vec{b})$)
   a. since the linear changes of coordinates involve the vertices, the **parameters appear in the exponents** of the generating functions,

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices$(P(\vec{b}))$ determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones$(P(\vec{b}))$ determines the vertex cones (= corners) of $P(\vec{b})$
   a. **Same challenges!**
   b. And, at the end, many sets of cases of the case discussion can be replaced by a single case, that is, doing **recombination**.
3. GeneratingFunction$(P(\vec{b}))$ determines the generating functions of each cone Cones$(P(\vec{b}))$
   a. since the linear changes of coordinates involve the vertices, the **parameters appear in the exponents** of the generating functions,
   b. thanks the periodicity of things, **quasi-polynomials** solve the issue.

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones($P(\vec{b})$) determines the vertex cones (= corners) of $P(\vec{b})$
   a. **Same challenges!**
   b. And, at the end, many sets of cases of the case discussion can be replaced by a single case, that is, doing **recombination**.
3. GeneratingFunction($P(\vec{b})$) determines the generating functions of each cone Cones($P(\vec{b})$)
   a. since the linear changes of coordinates involve the vertices, the **parameters appear in the exponents** of the generating functions,
   b. thanks the periodicity of things, **quasi-polynomials** solve the issue.
4. NumberOfIntegerPoints($P(\vec{b})$)

Integer point counting for parametric polyhedra

Given a parametric polyhedron $P(\vec{b})$, the procedures:

1. Vertices($P(\vec{b})$) determines the vertices of $P(\vec{b})$
   a. Yields to solve a (large) number of parametric linear systems, which are **independent** problems
   b. Their results need to be merged into a **single case discussion**
2. Cones($P(\vec{b})$) determines the vertex cones (= corners) of $P(\vec{b})$
   a. **Same challenges!**
   b. And, at the end, many sets of cases of the case discussion can be replaced by a single case, that is, doing **recombination**.
3. GeneratingFunction($P(\vec{b})$) determines the generating functions of each cone Cones($P(\vec{b})$)
   a. since the linear changes of coordinates involve the vertices, the **parameters appear in the exponents** of the generating functions,
   b. thanks the periodicity of things, **quasi-polynomials** solve the issue.
4. NumberOfIntegerPoints($P(\vec{b})$)
   a. Putting everything together requires computing with **multivariate quasi-polynomials.**

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets
2. Let $\mathcal{F}$ be a non-empty set of functions from $\mathcal{A}$ to $\mathcal{B}$.

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets
2. Let $\mathcal{F}$ be a non-empty set of functions from $\mathcal{A}$ to $\mathcal{B}$.
3. Let $\mathcal{P}$ be a non-empty set of predicates on $\mathcal{B}$. closed under negation.

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets
2. Let $\mathcal{F}$ be a non-empty set of functions from $\mathcal{A}$ to $\mathcal{B}$.
3. Let $\mathcal{P}$ be a non-empty set of predicates on $\mathcal{B}$. closed under negation.
4. A **constraint** is any pair $c = (f, p)$ where $f \in \mathcal{F}$ and $p \in \mathcal{P}$ and its zero set is
$$Z(c) = \{a \in \mathcal{A} \mid p(f(a))\} \tag{5.1}$$
while its negation is $\neg c := (f, \neg p)$.

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets
2. Let $\mathcal{F}$ be a non-empty set of functions from $\mathcal{A}$ to $\mathcal{B}$.
3. Let $\mathcal{P}$ be a non-empty set of predicates on $\mathcal{B}$. closed under negation.
4. A **constraint** is any pair $c = (f, p)$ where $f \in \mathcal{F}$ and $p \in \mathcal{P}$ and its zero set is
$$Z(c) = \{a \in \mathcal{A} \mid p(f(a))\} \tag{5.1}$$
while its negation is $\neg c := (f, \neg p)$.
5. The constraint $c = (f, p)$ is **consistent** whenever $Z(C) \neq \emptyset$ holds.

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets
2. Let $\mathcal{F}$ be a non-empty set of functions from $\mathcal{A}$ to $\mathcal{B}$.
3. Let $\mathcal{P}$ be a non-empty set of predicates on $\mathcal{B}$. closed under negation.
4. A **constraint** is any pair $c = (f, p)$ where $f \in \mathcal{F}$ and $p \in \mathcal{P}$ and its zero set is
$$Z(c) \ = \ \{a \in \mathcal{A} \ \mid \ p(f(a))\} \tag{5.1}$$
while its negation is $\neg c := (f, \neg p)$.
5. The constraint $c = (f, p)$ is **consistent** whenever $Z(C) \neq \emptyset$ holds.
6. A **system** of constraints is any finite set C of constraints and its zero set is
$$Z(C) \ = \ \bigcap_{c \in C} Z(c). \tag{5.2}$$

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets
2. Let $\mathcal{F}$ be a non-empty set of functions from $\mathcal{A}$ to $\mathcal{B}$.
3. Let $\mathcal{P}$ be a non-empty set of predicates on $\mathcal{B}$. closed under negation.
4. A **constraint** is any pair $c = (f, p)$ where $f \in \mathcal{F}$ and $p \in \mathcal{P}$ and its zero set is
$$Z(c) \ = \ \{a \in \mathcal{A} \ | \ p(f(a))\} \tag{5.1}$$
while its negation is $\neg c := (f, \neg p)$.
5. The constraint $c = (f, p)$ is **consistent** whenever $Z(C) \neq \varnothing$ holds.
6. A **system** of constraints is any finite set C of constraints and its zero set is
$$Z(C) \ = \ \bigcap_{c \in C} Z(c). \tag{5.2}$$
7. A constraint $\gamma \notin C$ is **redundant** w.r.t. C, whenever we have $Z(C \cup \{\gamma\}) \ = \ Z(C)$.

Generic case discussion (1/3)

1. Let $\mathcal{A}, \mathcal{B}, \mathcal{V}$ be 3 non-empty sets
2. Let $\mathcal{F}$ be a non-empty set of functions from $\mathcal{A}$ to $\mathcal{B}$.
3. Let $\mathcal{P}$ be a non-empty set of predicates on $\mathcal{B}$. closed under negation.
4. A **constraint** is any pair $c = (f, p)$ where $f \in \mathcal{F}$ and $p \in \mathcal{P}$ and its zero set is
$$Z(c) \; = \; \{a \in \mathcal{A} \;\mid\; p(f(a))\} \tag{5.1}$$
while its negation is $\neg c := (f, \neg p)$.
5. The constraint $c = (f, p)$ is **consistent** whenever $Z(C) \neq \varnothing$ holds.
6. A **system** of constraints is any finite set C of constraints and its zero set is
$$Z(C) \; = \; \bigcap_{c \in C} Z(c). \tag{5.2}$$
7. A constraint $\gamma \notin C$ is **redundant** w.r.t. C, whenever we have $Z(C \cup \{\gamma\}) \; = \; Z(C)$.
8. A **value-constraints** pair is any pair (V, C) where $V \subseteq \mathcal{V}$ and C is a system of constraints.

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \le i, j \le e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \leq i, j \leq e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.
3. S is **non-overlapping**, if, for all $1 \leq i < j \leq e$, we have $Z(C_i) \cap Z(C_j) = \emptyset$.

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \leq i, j \leq e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.
3. S is **non-overlapping**, if, for all $1 \leq i < j \leq e$, we have $Z(C_i) \cap Z(C_j) = \emptyset$.
4. Let $T = (W_1, D_1), \ldots, (W_f, D_f)$ be a second sequence of value-constraint pairs.
5. We say that T **refines** S whenever the following 3 properties all hold:

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \le i, j \le e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.
3. S is **non-overlapping**, if, for all $1 \le i < j \le e$, we have $Z(C_i) \cap Z(C_j) = \emptyset$.
4. Let $T = (W_1, D_1), \ldots, (W_f, D_f)$ be a second sequence of value-constraint pairs.
5. We say that T **refines** S whenever the following 3 properties all hold:
    a. we have: $\bigcup_{i=1}^{e} Z(C_i) = \bigcup_{i=1}^{f} Z(D_i)$,

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \leq i, j \leq e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.
3. S is **non-overlapping**, if, for all $1 \leq i < j \leq e$, we have $Z(C_i) \cap Z(C_j) = \emptyset$.
4. Let $T = (W_1, D_1), \ldots, (W_f, D_f)$ be a second sequence of value-constraint pairs.
5. We say that T **refines** S whenever the following 3 properties all hold:
   - a. we have: $\bigcup_{i=1}^{e} Z(C_i) = \bigcup_{i=1}^{f} Z(D_i)$,
   - b. we have: $\bigcup_{i=1}^{e} V_i = \bigcup_{i=1}^{f} W_i$,

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \le i, j \le e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.
3. S is **non-overlapping**, if, for all $1 \le i < j \le e$, we have $Z(C_i) \cap Z(C_j) = \emptyset$.
4. Let $T = (W_1, D_1), \ldots, (W_f, D_f)$ be a second sequence of value-constraint pairs.
5. We say that T **refines** S whenever the following 3 properties all hold:
   - a. we have: $\bigcup_{i=1}^{e} Z(C_i) = \bigcup_{i=1}^{f} Z(D_i)$,
   - b. we have: $\bigcup_{i=1}^{e} V_i = \bigcup_{i=1}^{f} W_i$,
   - c. $(\forall i,\ 1 \le i \le f)\ (\exists j, 1 \le j \le e)\quad Z(D_i) \subseteq Z(C_j)$ and $V_j \subseteq W_i$.

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \leq i, j \leq e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.
3. S is **non-overlapping**, if, for all $1 \leq i < j \leq e$, we have $Z(C_i) \cap Z(C_j) = \emptyset$.
4. Let $T = (W_1, D_1), \ldots, (W_f, D_f)$ be a second sequence of value-constraint pairs.
5. We say that T **refines** S whenever the following 3 properties all hold:
   - a. we have: $\bigcup_{i=1}^{e} Z(C_i) = \bigcup_{i=1}^{f} Z(D_i)$,
   - b. we have: $\bigcup_{i=1}^{e} V_i = \bigcup_{i=1}^{f} W_i$,
   - c. $(\forall i,\ 1 \leq i \leq f)\ (\exists j, 1 \leq j \leq e)\quad Z(D_i) \subseteq Z(C_j)$ and $V_j \subseteq W_i$.
6. We assume that we have a procedure that, for any system of constraints C, decides whether C is consistent or not.

Generic case discussion (2/3)

1. Let $S = (V_1, C_1), \ldots, (V_e, C_e)$ be a sequence of val.-constr. pairs.
2. S is **irredundant**, if, for all $1 \leq i, j \leq e$, we have $i \neq j \implies Z(C_i) \not\subseteq Z(C_j)$.
3. S is **non-overlapping**, if, for all $1 \leq i < j \leq e$, we have $Z(C_i) \cap Z(C_j) = \emptyset$.
4. Let $T = (W_1, D_1), \ldots, (W_f, D_f)$ be a second sequence of value-constraint pairs.
5. We say that T **refines** S whenever the following 3 properties all hold:
   - a. we have: $\bigcup_{i=1}^{e} Z(C_i) = \bigcup_{i=1}^{f} Z(D_i)$,
   - b. we have: $\bigcup_{i=1}^{e} V_i = \bigcup_{i=1}^{f} W_i$,
   - c. $(\forall i,\ 1 \leq i \leq f)\ (\exists j, 1 \leq j \leq e)\quad Z(D_i) \subseteq Z(C_j)$ and $V_j \subseteq W_i$.
6. We assume that we have a procedure that, for any system of constraints C, decides whether C is consistent or not.
7. Then, there exists an algorithm that, for the sequence S computes a non-overlapping sequence T refining S.

"Generic" case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.

“Generic” case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.

"Generic" case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.
3. Consider two systems of constraints C_1 and C_2

“Generic” case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.
3. Consider two systems of constraints C_1 and C_2
4. For each constraint $\gamma : p(\mathbf{x}) \geq 0$ of C_1

"Generic" case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.
3. Consider two systems of constraints C_1 and C_2
4. For each constraint $\gamma : p(\mathbf{x}) \geq 0$ of C_1
   a. γ is **valid over** C_2 if $p(\mathbf{x}) \geq 0$ for all $x \in Z(C_2)$

“Generic” case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.
3. Consider two systems of constraints C_1 and C_2
4. For each constraint $\gamma : p(\mathbf{x}) \geq 0$ of C_1
   a. γ is **valid over** C_2 if $p(\mathbf{x}) \geq 0$ for all $x \in Z(C_2)$
   b. γ is **separating over** C_2 if $p(\mathbf{x}) \leq -1$ for all $x \in Z(C_2)$

"Generic" case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.
3. Consider two systems of constraints C_1 and C_2
4. For each constraint $\gamma : p(\mathbf{x}) \geq 0$ of C_1
   a. γ is **valid over** C_2 if $p(\mathbf{x}) \geq 0$ for all $x \in Z(C_2)$
   b. γ is **separating over** C_2 if $p(\mathbf{x}) \leq -1$ for all $x \in Z(C_2)$
   c. γ is **cut over** C_2 if γ neither valid nor separating over C_2.

“Generic” case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.
3. Consider two systems of constraints C_1 and C_2
4. For each constraint $\gamma : p(\mathbf{x}) \geq 0$ of C_1
   a. γ is **valid over** C_2 if $p(\mathbf{x}) \geq 0$ for all $x \in Z(C_2)$
   b. γ is **separating over** C_2 if $p(\mathbf{x}) \leq -1$ for all $x \in Z(C_2)$
   c. γ is **cut over** C_2 if γ neither valid nor separating over C_2.
   d. If for $\gamma : p(\mathbf{x}) \geq 0$ of C_1 we have $p(\mathbf{x}) = -1 - u(\mathbf{x})$ and $u(\mathbf{x}) \geq 0$ is a constraint of C_2, then (p, u) is a pair of **adjacent** inequalities.

"Generic" case discussion (3/3)

1. Assume $\mathcal{A} = \mathcal{B} = \mathbb{Z}$ and $\mathcal{P} = \{\leq, \geq, \leq, \geq, =, \neq\}$.
2. Because $\mathcal{A} = \mathcal{B} = \mathbb{Z}$, we can normalize systems of constraints to use $\geq$ only.
3. Consider two systems of constraints C_1 and C_2
4. For each constraint $\gamma : p(\mathbf{x}) \geq 0$ of C_1
   a. γ is **valid over** C_2 if $p(\mathbf{x}) \geq 0$ for all $x \in Z(C_2)$
   b. γ is **separating over** C_2 if $p(\mathbf{x}) \leq -1$ for all $x \in Z(C_2)$
   c. γ is **cut over** C_2 if γ neither valid nor separating over C_2.
   d. If for $\gamma : p(\mathbf{x}) \geq 0$ of C_1 we have $p(\mathbf{x}) = -1 - u(\mathbf{x})$ and $u(\mathbf{x}) \geq 0$ is a constraint of C_2, then (p, u) is a pair of **adjacent** inequalities.
5. Theorem: If (p, u) is a pair of adjacent inequalities, and if all other constraints of C_1 (resp. C_2) are valid on C_2 (resp. C_1) then the system of constraints C_3 consisting of all those valid constraints satisfies $Z(C_3) = Z(C_1) \cup Z(C_2)$.

Concluding remarks

Summary and notes

1. We have presented Brion's formula and Barvinok's algorithm for computing the number of integer points of a polytope.
2. We have discussed our adaptation of those works to the case of parametric polyhedra and its implementation in MAPLE.
3. Another adaptation to this parametric case, tailored to compiler optimization, was led by Sven Verdoolaege and is part of a C library called `barvinok`.

Work in progress

1. Our MAPLE implementation aims at supporting Presburger arithmetic
2. This implementation is designed to extend to parametric polyhedra $\mathbf{A}\vec{x} \leq \vec{b}$ where parameters appear not only in $\vec{b}$ but also in $\mathbf{A}$.
3. Our current work focuses on minimizing the number of cases in the discussion and controlling expression swell.

Plan

Plan

Presburger arithmetic (1/2)

Definition 35

The language of **Presburger arithmetic** is:

1. the first-order theory of the integers with addition, equality and order
2. extended by the divisibility predicates $D_k : x \longmapsto k \mid x$, for all $k \in \mathbb{Z}_{>0}$.

For a more formal definition, see Wikipedia's **Presburger arithmetic**.

Presburger arithmetic (1/2)

Definition 35

The language of **Presburger arithmetic** is:

1. the first-order theory of the integers with addition, equality and order
2. extended by the divisibility predicates $D_k : x \longmapsto k \mid x$, for all $k \in \mathbb{Z}_{>0}$.

For a more formal definition, see Wikipedia's **Presburger arithmetic**.

Remark 3

A **Presburger formula F in prenex normal form** *has the form:*

$$F = Q_1 x_1 \cdots Q_m x_m \; \phi(x_1, \ldots, x_m, y_1, \ldots, y_n), \tag{6.1}$$

where:

1. $Q_1 x_1 \cdots Q_m x_m$ *is a sequence of quantifiers (existential or universal) and bound variables,*
2. $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ *is a quantifier-free formula,*
3. $y_1, \ldots, y_n$ *are free (or unbounded) variables.*

Presburger arithmetic (1/2)

Remark 4

1. *We shall assume that any quantifier-free formula $F(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is in* **disjunctive normal form** *(DNF).*
2. *Hence, it has the form:*

$$\phi(x_1, \ldots, x_m, y_1, \ldots, y_n) \; = \; \bigvee_i \bigwedge_j \; \Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n), \quad (6.2)$$

   where:

   a. *each $\Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is an* **atomic formula** *(or* **atom***),*
   b. *thus a formula free of quantifiers and connectives.*

Presburger arithmetic (1/2)

Remark 4

1. *We shall assume that any quantifier-free formula $F(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is in* **disjunctive normal form** *(DNF).*
2. *Hence, it has the form:*

$$\phi(x_1, \ldots, x_m, y_1, \ldots, y_n) = \bigvee_i \bigwedge_j \Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n), \quad (6.2)$$

   where:
   - a. *each $\Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is an* **atomic formula** *(or* **atom***),*
   - b. *thus a formula free of quantifiers and connectives.*

Remark 5

We can assume that each atom is either

1. *a non-strict inequality $\ell(x_1, \ldots, x_m, y_1, \ldots, y_n) \leq 0$,*
2. *or a divisibility relation $k \mid \ell(x_1, \ldots, x_m, y_1, \ldots, y_n)$,*

where

Presburger arithmetic (1/2)

Remark 4

1. *We shall assume that any quantifier-free formula $F(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is in* **disjunctive normal form** *(DNF).*
2. *Hence, it has the form:*

$$\phi(x_1, \ldots, x_m, y_1, \ldots, y_n) = \bigvee_i \bigwedge_j \Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n), \quad (6.2)$$

   where:

   a. *each $\Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is an* **atomic formula** *(or* **atom***),*
   b. *thus a formula free of quantifiers and connectives.*

Remark 5

We can assume that each atom is either

1. *a non-strict inequality $\ell(x_1, \ldots, x_m, y_1, \ldots, y_n) \leq 0$,*
2. *or a divisibility relation $k \mid \ell(x_1, \ldots, x_m, y_1, \ldots, y_n)$,*

where

1. *$\ell(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is a* **linear** *polynomial, that is, with total degree at most 1, in the variables $x_1, \ldots, x_m, y_1, \ldots, y_n$, and with integer coefficients.*

Presburger arithmetic (1/2)

Remark 4

1. *We shall assume that any quantifier-free formula $F(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is in* **disjunctive normal form** *(DNF).*
2. *Hence, it has the form:*

$$\phi(x_1, \ldots, x_m, y_1, \ldots, y_n) \;=\; \bigvee_i \bigwedge_j \; \Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n), \quad (6.2)$$

   where:
   - a. *each $\Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is an* **atomic formula** *(or* **atom***),*
   - b. *thus a formula free of quantifiers and connectives.*

Remark 5

We can assume that each atom is either

1. *a non-strict inequality $\ell(x_1, \ldots, x_m, y_1, \ldots, y_n) \leq 0$,*
2. *or a divisibility relation $k \mid \ell(x_1, \ldots, x_m, y_1, \ldots, y_n)$,*

where

1. *$\ell(x_1, \ldots, x_m, \, y_1, \ldots, y_n)$ is a* **linear** *polynomial, that is, with total degree at most 1, in the variables $x_1, \ldots, x_m, y_1, \ldots, y_n$, and with integer coefficients.*
2. *Of course, each variable is meant to take values in $\mathbb{Z}$.*

Quantifier elimination (1/2)

Theorem 36

Presburger arithmetic admits quantifier elimination.

Proof.

1. See the thesis of Mojżesz Presburger [20] the paper of David Cooper [4], and Christoph Haase's **Survival Guide to Presburger Arithmetic** [7].
2. See also our own proof in a few slides.

□

Quantifier elimination (1/2)

Theorem 36

Presburger arithmetic admits quantifier elimination.

Proof.

1. See the thesis of Mojżesz Presburger [20] the paper of David Cooper [4], and Christoph Haase's **Survival Guide to Presburger Arithmetic** [7].
2. See also our own proof in a few slides.

□

Remark 6

Therefore, our goal is to determine the set $D(y_1, \ldots, y_n) \subseteq \mathbb{Z}^n$ *of* **ALL** *integer tuples of* $(y_1, \ldots, y_n)$ *for which the formula* $F(x_1, \ldots, x_m, y_1, \ldots, y_n)$ *is true.*

Quantifier elimination (2/2)

Remark 7

1. *Recall*

$$F = Q_1 x_1 \cdots Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

Quantifier elimination (2/2)

Remark 7

1. *Recall*

$$F = Q_1 x_1 \cdots Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

2. *If $m = 0$, then it "suffices" to determine the tuples of integer values $(y_1, \ldots, y_n)$ for which $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is true.*

Quantifier elimination (2/2)

Remark 7

1. *Recall*

$$F = Q_1 x_1 \cdots Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

2. *If* $m = 0$*, then it "suffices" to determine the tuples of integer values* $(y_1, \ldots, y_n)$ *for which* $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ *is true.*
3. *Suppose* $m > 0$*. By induction, assume also* $F = Q x_1 F'$*, where* F' *is quantifier-free.*

Quantifier elimination (2/2)

Remark 7

1. *Recall*

$$F = Q_1 x_1 \cdots Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

2. *If $m = 0$, then it "suffices" to determine the tuples of integer values $(y_1, \ldots, y_n)$ for which $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is true.*
3. *Suppose $m > 0$. By induction, assume also $F = Qx_1 F'$, where F' is quantifier-free.*
    a. *If $Q = \exists$, then we are now dealing with* **integer projection**, *see next section.*

Quantifier elimination (2/2)

Remark 7

1. *Recall*

$$F = Q_1 x_1 \cdots Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

2. *If $m = 0$, then it "suffices" to determine the tuples of integer values $(y_1, \ldots, y_n)$ for which $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is true.*
3. *Suppose $m > 0$. By induction, assume also $F = Qx_1 F'$, where F' is quantifier-free.*
   a. *If $Q = \exists$, then we are now dealing with* **integer projection***, see next section.*
   b. *If $Q = \forall$, then we can replace $\forall x_1 \ F'$ with $\neg(\exists x_1 \ \neg(F'))$,*

Quantifier elimination (2/2)

Remark 7

1. *Recall*

$$F = Q_1 x_1 \cdots Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

2. *If $m = 0$, then it "suffices" to determine the tuples of integer values $(y_1, \ldots, y_n)$ for which $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is true.*
3. *Suppose $m > 0$. By induction, assume also $F = Qx_1 F'$, where F' is quantifier-free.*
   1. *If $Q = \exists$, then we are now dealing with* **integer projection**, *see next section.*
   2. *If $Q = \forall$, then we can replace $\forall x_1 \ F'$ with $\neg(\exists x_1 \ \neg(F'))$,*
   3. *Whenever possible, we should make use of rules like:*

$$\forall x_1 \cdots \forall x_m \ C \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} = \mathbf{q} \ \Rightarrow \ C = \mathbf{0} \bigwedge \mathbf{q} = \mathbf{0}, \qquad (6.3)$$

   where

      1. *$C \in \mathbb{Z}^{r \times m}$ is a matrix, and*

Quantifier elimination (2/2)

Remark 7

1. *Recall*
$$F = Q_1 x_1 \cdots Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n),$$
2. *If $m = 0$, then it "suffices" to determine the tuples of integer values $(y_1, \ldots, y_n)$ for which $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is true.*
3. *Suppose $m > 0$. By induction, assume also $F = Q x_1 F'$, where F' is quantifier-free.*
   a. *If $Q = \exists$, then we are now dealing with* **integer projection**, *see next section.*
   b. *If $Q = \forall$, then we can replace $\forall x_1 \ F'$ with $\neg(\exists x_1 \ \neg(F'))$,*
   c. *Whenever possible, we should make use of rules like:*

$$\forall x_1 \cdots \forall x_m \ C \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} = \mathbf{q} \ \Rightarrow \ C = \mathbf{0} \bigwedge \mathbf{q} = \mathbf{0}, \qquad (6.3)$$

   where

   1. *$C \in \mathbb{Z}^{r \times m}$ is a matrix, and*
   2. *$\mathbf{q} \in (\mathbb{Z}[y_1, \ldots, y_m])^r$ is a vector of linear polynomials.*

Coarsening the atoms

Remark 8

In Cooper's algorithm, when processing $\exists x_1\ F'$, the formula F' uses the following four types of atoms:

$$A_y < ax_1,\ ax_1 < A_y,\ k \mid (ax_1 + A_y),\quad \text{and}\quad \neg (k \mid (ax_1 + A_y))\,, \tag{6.4}$$

where $a \in \mathbb{Z}$ and $A_y \in \mathbb{Z}[y_1, \ldots, y_n]$ is a linear polynomial.

Coarsening the atoms

Remark 8

In Cooper's algorithm, when processing $\exists x_1\ F'$, the formula F' uses the following four types of atoms:

$$A_y < ax_1,\ ax_1 < A_y,\ k \mid (ax_1 + A_y),\ \ \text{and}\ \ \neg\,(k \mid (ax_1 + A_y))\,, \qquad (6.4)$$

where $a \in \mathbb{Z}$ and $A_y \in \mathbb{Z}[y_1, \ldots, y_n]$ is a linear polynomial.

Remark 9

We can rearrange our quantifier-free formula to:

$$\phi(x_1, \ldots, x_m, y_1, \ldots, y_n) \ = \ \bigvee_i \ Z_i(x_1, \ldots, x_m, y_1, \ldots, y_n), \qquad (6.5)$$

where each Z_i is a predicate of the form

$$\begin{pmatrix} x_1 \\ \vdots \\ x_m \\ y_1 \\ \vdots \\ y_m \end{pmatrix} \in \ \mathbb{Z}\text{Polyhedron}(P_i, L_i), \qquad (6.6)$$

for some polyhedra P_i and integer lattices L_i.
We call such a predicate a **$\mathbb{Z}$-polyhedron predicate**.

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,
2. let $N \in \mathbb{Z}^{r \times n}$ be an integer matrix,

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,
2. let $N \in \mathbb{Z}^{r \times n}$ be an integer matrix,
3. let $\mathbf{z}$ be an n-dimensional column vector whose coordinates are n independent integral variables $z_1, \ldots, z_n$,

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,
2. let $N \in \mathbb{Z}^{r \times n}$ be an integer matrix,
3. let **z** be an n-dimensional column vector whose coordinates are n independent integral variables $z_1, \ldots, z_n$,
4. let **q** be an r-dimensional column vector whose coordinates are linear polynomials $q_1, \ldots, q_r \in \mathbb{Z}[w_1, \ldots, w_\nu]$,

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,
2. let $N \in \mathbb{Z}^{r \times n}$ be an integer matrix,
3. let $\mathbf{z}$ be an n-dimensional column vector whose coordinates are n independent integral variables $z_1, \ldots, z_n$,
4. let $\mathbf{q}$ be an r-dimensional column vector whose coordinates are linear polynomials $q_1, \ldots, q_r \in \mathbb{Z}[w_1, \ldots, w_\nu]$,
5. we regard the variables $\mathbf{w} = w_1, \ldots, w_\nu$ as parameters,

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,
2. let $N \in \mathbb{Z}^{r \times n}$ be an integer matrix,
3. let $\mathbf{z}$ be an n-dimensional column vector whose coordinates are n independent integral variables $z_1, \ldots, z_n$,
4. let $\mathbf{q}$ be an r-dimensional column vector whose coordinates are linear polynomials $q_1, \ldots, q_r \in \mathbb{Z}[w_1, \ldots, w_\nu]$,
5. we regard the variables $\mathbf{w} = w_1, \ldots, w_\nu$ as parameters,
6. let $\mathbf{m} \in {\mathbb{Z}_{>0}}^r$.

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,
2. let $N \in \mathbb{Z}^{r \times n}$ be an integer matrix,
3. let $\mathbf{z}$ be an n-dimensional column vector whose coordinates are n independent integral variables $z_1, \ldots, z_n$,
4. let $\mathbf{q}$ be an r-dimensional column vector whose coordinates are linear polynomials $q_1, \ldots, q_r \in \mathbb{Z}[w_1, \ldots, w_\nu]$,
5. we regard the variables $\mathbf{w} = w_1, \ldots, w_\nu$ as parameters,
6. let $\mathbf{m} \in {\mathbb{Z}_{>0}}^r$.
7. Consider the system

$$N\,\mathbf{z} \equiv \mathbf{q} \mod \mathbf{m}. \tag{6.7}$$

Solving parametric systems of linear congruences

1. Let $r, n \in \mathbb{Z}_{>0}$,
2. let $N \in \mathbb{Z}^{r \times n}$ be an integer matrix,
3. let $\mathbf{z}$ be an n-dimensional column vector whose coordinates are n independent integral variables $z_1, \ldots, z_n$,
4. let $\mathbf{q}$ be an r-dimensional column vector whose coordinates are linear polynomials $q_1, \ldots, q_r \in \mathbb{Z}[w_1, \ldots, w_\nu]$,
5. we regard the variables $\mathbf{w} = w_1, \ldots, w_\nu$ as parameters,
6. let $\mathbf{m} \in {\mathbb{Z}_{>0}}^r$.
7. Consider the system

$$N\,\mathbf{z} \equiv \mathbf{q} \mod \mathbf{m}. \tag{6.7}$$

Theorem 37 (parametric multivariate CRT)

The values of $(w_1, \ldots, w_\nu)$ for which the above system has solutions form a lattice of $\mathbb{Z}^\nu$. Moreover, for each value of $(w_1, \ldots, w_\nu)$, the $\mathbf{z}$*-solutions form a lattice of $\mathbb{Z}^n$.*

Proof.

Compute the Hermite normal forms of the appropriate matrices. □

Plan

Integer projection: $n = 1$

Remark 10

1. *From the above section, we consider the formula*

$$\exists x \; \phi(x, y_1, \ldots, y_n), \quad \text{where} \quad \phi(x, y_1, \ldots, y_n) = \bigvee_i \phi_i(x, y_1, \ldots, y_n), \tag{6.8}$$

where $\phi_i(x, y_1, \ldots, y_n)$ is a conjunction of congruence relations and non-strict inequalities.

Integer projection: $n = 1$

Remark 10

1. *From the above section, we consider the formula*
$$\exists x\ \phi(x, y_1, \ldots, y_n), \quad \text{where} \quad \phi(x, y_1, \ldots, y_n) = \bigvee_i \phi_i(x, y_1, \ldots, y_n), \tag{6.8}$$
where $\phi_i(x, y_1, \ldots, y_n)$ is a conjunction of congruence relations and non-strict inequalities.
2. *We want the values of $\mathbf{y} = y_1, \ldots y_n$ so that $\exists x\ \phi(x, y_1, \ldots, y_n)$ holds.*

Integer projection: $n = 1$

Remark 10

1. *From the above section, we consider the formula*
$$\exists x\ \phi(x, y_1, \ldots, y_n), \ \text{where}\ \phi(x, y_1, \ldots, y_n) = \bigvee_i \phi_i(x, y_1, \ldots, y_n), \tag{6.8}$$
where $\phi_i(x, y_1, \ldots, y_n)$ is a conjunction of congruence relations and non-strict inequalities.
2. *We want the values of $\mathbf{y} = y_1, \ldots y_n$ so that $\exists x\ \phi(x, y_1, \ldots, y_n)$ holds.*
3. *We can further reduce the problem as follows.*

Integer projection: $n = 1$

Remark 10

1. *From the above section, we consider the formula*
$$\exists x\ \phi(x, y_1, \ldots, y_n), \quad \text{where} \quad \phi(x, y_1, \ldots, y_n) = \bigvee_i \phi_i(x, y_1, \ldots, y_n), \tag{6.8}$$
where $\phi_i(x, y_1, \ldots, y_n)$ is a conjunction of congruence relations and non-strict inequalities.
2. *We want the values of $\mathbf{y} = y_1, \ldots y_n$ so that $\exists x\ \phi(x, y_1, \ldots, y_n)$ holds.*
3. *We can further reduce the problem as follows.*

Remark 11

1. *Let $f_1, \ldots, f_s$, $g_1, \ldots, g_r \in \mathbb{Z}[x, \mathbf{y}]$ be linear and let $k_1, \ldots, k_r \in \mathbb{Z}_{>0}$.*

Integer projection: $n = 1$

Remark 10

1. *From the above section, we consider the formula*
$$\exists x\ \phi(x, y_1, \ldots, y_n), \quad \text{where} \quad \phi(x, y_1, \ldots, y_n) = \bigvee_i \ \phi_i(x, y_1, \ldots, y_n), \tag{6.8}$$
where $\phi_i(x, y_1, \ldots, y_n)$ *is a conjunction of congruence relations and non-strict inequalities.*
2. *We want the values of* $\mathbf{y} = y_1, \ldots y_n$ *so that* $\exists x\ \phi(x, y_1, \ldots, y_n)$ *holds.*
3. *We can further reduce the problem as follows.*

Remark 11

1. *Let* $f_1, \ldots, f_s,\ g_1, \ldots, g_r \in \mathbb{Z}[x, \mathbf{y}]$ *be linear and let* $k_1, \ldots, k_r \in \mathbb{Z}_{>0}$.
2. *Consider the formula:*
$$F(\mathbf{y}) : (\exists x \in \mathbb{Z}) \left\{ \begin{array}{ccc} f_1 & \leq & 0 \\ \vdots & \vdots & \vdots \\ f_s & \leq & 0 \end{array} \right. \wedge \begin{array}{cccc} g_1 & \equiv & 0 & \bmod k_1 \\ \vdots & \vdots & \vdots & \\ g_r & \equiv & 0 & \bmod k_r \end{array} \tag{6.9}$$

Integer projection: $n = 1$

Remark 10

1. *From the above section, we consider the formula*
$$\exists x\ \phi(x, y_1, \ldots, y_n), \quad \text{where} \quad \phi(x, y_1, \ldots, y_n) = \bigvee_i \phi_i(x, y_1, \ldots, y_n), \tag{6.8}$$
where $\phi_i(x, y_1, \ldots, y_n)$ is a conjunction of congruence relations and non-strict inequalities.
2. *We want the values of $\mathbf{y} = y_1, \ldots y_n$ so that $\exists x\ \phi(x, y_1, \ldots, y_n)$ holds.*
3. *We can further reduce the problem as follows.*

Remark 11

1. *Let $f_1, \ldots, f_s$, $g_1, \ldots, g_r \in \mathbb{Z}[x, \mathbf{y}]$ be linear and let $k_1, \ldots, k_r \in \mathbb{Z}_{>0}$.*
2. *Consider the formula:*
$$F(\mathbf{y}) : (\exists x \in \mathbb{Z}) \left\{ \begin{array}{ccc} f_1 & \leq & 0 \\ \vdots & \vdots & \vdots \\ f_s & \leq & 0 \end{array} \right. \wedge \begin{array}{cccc} g_1 & \equiv & 0 & \bmod k_1 \\ \vdots & \vdots & \vdots & \\ g_r & \equiv & 0 & \bmod k_r \end{array} \tag{6.9}$$
3. *We shall determine the set $D(\mathbf{y})$ of integer tuples $(y_1, \ldots y_n)$ for which $F(\mathbf{y})$ holds. We call $D(\mathbf{y})$ the* **integer projection** *of $F(\mathbf{y})$.*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*
4. *We should also check for implicit equations.*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*
4. *We should also check for implicit equations.*
5. *This process*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*
4. *We should also check for implicit equations.*
5. *This process*
   a. *introduces new variables (in order to define the lattice),*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*
4. *We should also check for implicit equations.*
5. *This process*
    a. *introduces new variables (in order to define the lattice),*
    b. *but eliminates at least the same number of variables from our system of linear inequalities*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null*.*
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*
4. *We should also check for implicit equations.*
5. *This process*
   - a. *introduces new variables (in order to define the lattice),*
   - b. *but eliminates at least the same number of variables from our system of linear inequalities*
6. *As a result, we now have a $\mathbb{Z}$-polyhedron predicate.*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*
4. *We should also check for implicit equations.*
5. *This process*
   a. *introduces new variables (in order to define the lattice),*
   b. *but eliminates at least the same number of variables from our system of linear inequalities*
6. *As a result, we now have a $\mathbb{Z}$-polyhedron predicate.*
7. *If that process* **solves for x**, *then our problem becomes that of describing the points of a $\mathbb{Z}$-polyhedron, which can be done by our integer hull algorithm.*

Integer projection: $n = 1$, removing congruences

Remark 12

1. *Suppose that $r > 0$ holds, that is, we do have congruences.*
2. *We apply Theorem 37* null.
3. *Now some of $x, y_1, \ldots, y_n$ are given by a lattice.*
4. *We should also check for implicit equations.*
5. *This process*
   a. *introduces new variables (in order to define the lattice),*
   b. *but eliminates at least the same number of variables from our system of linear inequalities*
6. *As a result, we now have a $\mathbb{Z}$-polyhedron predicate.*
7. *If that process* **solves for** x*, then our problem becomes that of describing the points of a $\mathbb{Z}$-polyhedron, which can be done by our integer hull algorithm.*
8. *If that process* **does not solve for** x*, then go to next slide.*

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*
2. *We start with the case $s = 2$ and rename f_1, f_2 to A, B.*

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*
2. *We start with the case $s = 2$ and rename f_1, f_2 to A, B.*
3. *We also write:*

$$A = A_{\mathbf{y}} - a\,x, \quad \text{and} \quad B = -B_{\mathbf{y}} + b\,x, \tag{6.10}$$

where $a, b \in \mathbb{Z}$ are non-zero and where $A_{\mathbf{y}}, B_{\mathbf{y}} \in \mathbb{Z}[\mathbf{y}]$ are linear

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*
2. *We start with the case $s = 2$ and rename f_1, f_2 to A, B.*
3. *We also write:*
$$A = A_{\mathbf{y}} - a\,x, \quad \text{and} \quad B = -B_{\mathbf{y}} + b\,x, \tag{6.10}$$
where $a, b \in \mathbb{Z}$ are non-zero and where $A_{\mathbf{y}}, B_{\mathbf{y}} \in \mathbb{Z}[\mathbf{y}]$ are linear
4. *We further assume that $a > 0$ and $b > 0$ both hold.*

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*
2. *We start with the case $s = 2$ and rename f_1, f_2 to A, B.*
3. *We also write:*
$$A = A_{\mathbf{y}} - a\,x, \quad \text{and} \quad B = -B_{\mathbf{y}} + b\,x, \tag{6.10}$$
where $a, b \in \mathbb{Z}$ are non-zero and where $A_{\mathbf{y}}, B_{\mathbf{y}} \in \mathbb{Z}[\mathbf{y}]$ are linear
4. *We further assume that $a > 0$ and $b > 0$ both hold.*
5. *With these assumptions, we call the inequalities $A \leq 0$ and $B \leq 0$, respectively a* **lower bound** *and an* **upper bound** *for x.*

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*
2. *We start with the case* $s = 2$ *and rename* f_1, f_2 *to* A, B.
3. *We also write:*
$$A = A_{\mathbf{y}} - a\,x, \quad \text{and} \quad B = -B_{\mathbf{y}} + b\,x, \tag{6.10}$$
where $a, b \in \mathbb{Z}$ *are non-zero and where* $A_{\mathbf{y}}, B_{\mathbf{y}} \in \mathbb{Z}[\mathbf{y}]$ *are linear*
4. *We further assume that* $a > 0$ *and* $b > 0$ *both hold.*
5. *With these assumptions, we call the inequalities* $A \leq 0$ *and* $B \leq 0$, *respectively a* **lower bound** *and an* **upper bound** *for* x.
6. *Observe that Formula (6.9* null*) simplifies to:*
$$F(\mathbf{y}) : (\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \leq a\,x)\ \wedge\ (b\,x \leq B_{\mathbf{y}}), \tag{6.11}$$

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*
2. *We start with the case $s = 2$ and rename f_1, f_2 to A, B.*
3. *We also write:*
$$A = A_{\mathbf{y}} - a\,x, \quad \text{and} \quad B = -B_{\mathbf{y}} + b\,x, \tag{6.10}$$
where $a, b \in \mathbb{Z}$ are non-zero and where $A_{\mathbf{y}}, B_{\mathbf{y}} \in \mathbb{Z}[\mathbf{y}]$ are linear
4. *We further assume that $a > 0$ and $b > 0$ both hold.*
5. *With these assumptions, we call the inequalities $A \le 0$ and $B \le 0$, respectively a* **lower bound** *and an* **upper bound** *for x.*
6. *Observe that Formula (6.9* null*) simplifies to:*
$$F(\mathbf{y}) : (\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \le a\,x)\ \wedge\ (b\,x \le B_{\mathbf{y}}), \tag{6.11}$$
7. *We present a first formula for $D(\mathbf{y})$ based on Harris Williams [23, 24]*

Integer projection: $n = 1$, no congruences, 2 inequalities

Remark 13

1. *We have used the congruences and we can focus on the inequalities.*
2. *We start with the case $s = 2$ and rename f_1, f_2 to A, B.*
3. *We also write:*
$$A = A_{\mathbf{y}} - a\,x, \quad \text{and} \quad B = -B_{\mathbf{y}} + b\,x, \tag{6.10}$$
where $a, b \in \mathbb{Z}$ are non-zero and where $A_{\mathbf{y}}, B_{\mathbf{y}} \in \mathbb{Z}[\mathbf{y}]$ are linear
4. *We further assume that $a > 0$ and $b > 0$ both hold.*
5. *With these assumptions, we call the inequalities $A \leq 0$ and $B \leq 0$, respectively a* **lower bound** *and an* **upper bound** *for x.*
6. *Observe that Formula (6.9 null) simplifies to:*
$$F(\mathbf{y}) : (\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \leq a\,x)\ \wedge\ (b\,x \leq B_{\mathbf{y}}), \tag{6.11}$$
7. *We present a first formula for $D(\mathbf{y})$ based on Harris Williams [23, 24]*
8. *Then, we present a second one based on William Pugh's Omega test [16, 17].*

Williams-style Projection

Theorem 38

Let $\ell = \mathrm{lcm}(a, b)$*,* $b' = \ell/a$ *and* $a' = \ell/b$*. For* $0 \le k < b$*, define*

$$E_k := \{\mathbf{y} \mid \mathrm{rem}(B_{\mathbf{y}}, b) = k\}.$$

Then, the following two formulas are equivalent:

1. $F(\mathbf{y})$*:* $(\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \le a\,x) \ \wedge\ (b\,x \le B_{\mathbf{y}})$,
2. $\bigvee_{k=0}^{k=b-1} (\mathbf{y} \in E_k) \wedge (a'k \ \le\ a' B_{\mathbf{y}} - b' A_{\mathbf{y}})$.

Williams-style Projection

Theorem 38

Let $\ell = \mathrm{lcm}(a, b)$*,* $b' = \ell/a$ *and* $a' = \ell/b$*. For* $0 \le k < b$*, define*

$$E_k := \{\mathbf{y} \mid \mathrm{rem}(B_{\mathbf{y}}, b) = k\}.$$

Then, the following two formulas are equivalent:

1. $F(\mathbf{y})$: $(\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \le a\,x) \ \wedge\ (b\,x \le B_{\mathbf{y}})$,
2. $\bigvee_{k=0}^{k=b-1} (\mathbf{y} \in E_k) \wedge (a'k \ \le\ a' B_{\mathbf{y}} - b' A_{\mathbf{y}})$.

Proof.

1. If $a = 1$ or $b = 1$ holds, then: $F(\mathbf{y}) \iff b'\,A_{\mathbf{y}} \le a'\,B_{\mathbf{y}}$.

□

Williams-style Projection

Theorem 38

Let $\ell = \mathrm{lcm}(a, b)$, $b' = \ell/a$ and $a' = \ell/b$. For $0 \le k < b$, define

$$E_k := \{\mathbf{y} \mid \mathrm{rem}(B_{\mathbf{y}}, b) = k\}.$$

Then, the following two formulas are equivalent:

1. *F(**y**):* $(\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \le a\,x)\ \wedge\ (b\,x \le B_{\mathbf{y}})$,
2. $\bigvee_{k=0}^{k=b-1} (\mathbf{y} \in E_k) \wedge (a'k \le a' B_{\mathbf{y}} - b' A_{\mathbf{y}})$.

Proof.

1. If $a = 1$ or $b = 1$ holds, then: $F(\mathbf{y}) \iff b'\,A_{\mathbf{y}} \le a'\,B_{\mathbf{y}}$.
2. From now on, assume $a > 1$ and $b > 1$ both hold. Observe that

$$F(\mathbf{y}) \iff (\exists x \in \mathbb{Z})\ (b' A_{\mathbf{y}} \le \ell\,x)\ \wedge\ (\ell\,x \le a' B_{\mathbf{y}}).$$

□

Williams-style Projection

Theorem 38

Let $\ell = \mathrm{lcm}(a, b)$*,* $b' = \ell/a$ *and* $a' = \ell/b$*. For* $0 \le k < b$*, define*

$$E_k := \{\mathbf{y} \mid \mathrm{rem}(B_{\mathbf{y}}, b) = k\}.$$

Then, the following two formulas are equivalent:

1. $F(\mathbf{y})$: $(\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \le a\,x) \ \wedge\ (b\,x \le B_{\mathbf{y}})$,
2. $\bigvee_{k=0}^{k=b-1} (\mathbf{y} \in E_k) \wedge (a'k \ \le\ a'B_{\mathbf{y}} - b'A_{\mathbf{y}})$.

Proof.

1. If $a = 1$ or $b = 1$ holds, then: $F(\mathbf{y}) \iff b'\,A_{\mathbf{y}} \le a'\,B_{\mathbf{y}}$.
2. From now on, assume $a > 1$ and $b > 1$ both hold. Observe that
$$F(\mathbf{y}) \iff (\exists x \in \mathbb{Z})\ (b'A_{\mathbf{y}} \le \ell\,x) \ \wedge\ (\ell\,x \le a'B_{\mathbf{y}}).$$
3. Hence, $F(\mathbf{y})$ says that a multiple of ℓ lies between $b'A_{\mathbf{y}}$ and $a'B_{\mathbf{y}}$).

□

Williams-style Projection

Theorem 38

Let $\ell = \operatorname{lcm}(a,b)$, $b' = \ell/a$ *and* $a' = \ell/b$. *For* $0 \le k < b$, *define*

$$E_k := \{\mathbf{y} \mid \operatorname{rem}(B_{\mathbf{y}}, b) = k\}.$$

Then, the following two formulas are equivalent:

1. *F(y):* $(\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \le a\,x)\ \wedge\ (b\,x \le B_{\mathbf{y}})$,
2. $\bigvee_{k=0}^{k=b-1}(\mathbf{y} \in E_k) \wedge (a'k \ \le\ a'B_{\mathbf{y}} - b'A_{\mathbf{y}})$.

Proof.

1. If $a = 1$ or $b = 1$ holds, then: $F(\mathbf{y}) \iff b'\,A_{\mathbf{y}} \le a'\,B_{\mathbf{y}}$.
2. From now on, assume $a > 1$ and $b > 1$ both hold. Observe that
$$F(\mathbf{y}) \iff (\exists x \in \mathbb{Z})\ (b'A_{\mathbf{y}} \le \ell\,x)\ \wedge\ (\ell\,x \le a'B_{\mathbf{y}}).$$
3. Hence, $F(\mathbf{y})$ says that a multiple of ℓ lies between $b'A_{\mathbf{y}}$ and $a'B_{\mathbf{y}}$).
4. Thus, $F(\mathbf{y}) \iff b'\,A_{\mathbf{y}} \ \le\ a'\,B_{\mathbf{y}} - \operatorname{rem}(a'\,B_{\mathbf{y}}, \ell)$.
5. That is, $F(\mathbf{y}) \iff b'A_{\mathbf{y}} \ \le\ a'(B_{\mathbf{y}} - \operatorname{rem}(B_{\mathbf{y}}, b))$.

□

Pugh's omega test (1/2)

Lemma 39 (William Pugh)

If we have:

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \ \geq \ (a-1)(b-1) \tag{6.12}$$

then $F(\mathbf{y})$ *holds.*

Pugh's omega test (1/2)

Lemma 39 (William Pugh)

If we have:

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \;\geq\; (a-1)(b-1) \tag{6.12}$$

then $F(\mathbf{y})$ holds.

Proof.

1. Consider the closed interval: $I \;:=\; \left(\frac{A_{\mathbf{y}}}{a}, \frac{B_{\mathbf{y}}}{b}\right)$.

Pugh's omega test (1/2)

Lemma 39 (William Pugh)

If we have:

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \;\geq\; (a-1)(b-1) \tag{6.12}$$

then $F(\mathbf{y})$ holds.

Proof.

1. Consider the closed interval: $I \;:=\; \left(\frac{A_{\mathbf{y}}}{a}, \frac{B_{\mathbf{y}}}{b}\right)$.
2. If I does not contain an integer, then we have:

$$i < \frac{A_{\mathbf{y}}}{a} \leq \frac{B_{\mathbf{y}}}{b} < i+1, \quad \text{where} \quad i \;=\; \left\lfloor \frac{A_{\mathbf{y}}}{a} \right\rfloor. \tag{6.13}$$

Pugh's omega test (1/2)

Lemma 39 (William Pugh)

If we have:

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \ \geq \ (a-1)(b-1) \tag{6.12}$$

then $F(\mathbf{y})$ *holds.*

Proof.

1. Consider the closed interval: $I \ := \ \left(\frac{A_{\mathbf{y}}}{a}, \frac{B_{\mathbf{y}}}{b}\right)$.
2. If I does not contain an integer, then we have:
$$i < \frac{A_{\mathbf{y}}}{a} \leq \frac{B_{\mathbf{y}}}{b} < i+1, \quad \text{where} \quad i \ = \ \left\lfloor \frac{A_{\mathbf{y}}}{a} \right\rfloor. \tag{6.13}$$
3. Let $\rho := \mathrm{rem}(A_{\mathbf{y}}, a)$. Since $i < \frac{A_{\mathbf{y}}}{a}$ holds, we have:
$$A_{\mathbf{y}} \ = \ i\,a + \rho \ \ \text{and} \ 0 < \rho < a, \tag{6.14}$$
4. from which we deduce: $\frac{A_{\mathbf{y}}}{a} - i \ \geq \ \frac{1}{a}$.

Pugh's omega test (1/2)

Lemma 39 (William Pugh)

If we have:

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \ \geq \ (a-1)(b-1) \tag{6.12}$$

then $F(\mathbf{y})$ holds.

Proof.

1. Consider the closed interval: $I \ := \ \left(\frac{A_{\mathbf{y}}}{a}, \frac{B_{\mathbf{y}}}{b}\right)$.
2. If I does not contain an integer, then we have:
$$i < \frac{A_{\mathbf{y}}}{a} \leq \frac{B_{\mathbf{y}}}{b} < i+1, \quad \text{where} \quad i \ = \ \left\lfloor \frac{A_{\mathbf{y}}}{a} \right\rfloor. \tag{6.13}$$
3. Let $\rho := \mathrm{rem}(A_{\mathbf{y}}, a)$. Since $i < \frac{A_{\mathbf{y}}}{a}$ holds, we have:
$$A_{\mathbf{y}} \ = \ i\,a + \rho \ \text{ and } 0 < \rho < a, \tag{6.14}$$
4. from which we deduce: $\frac{A_{\mathbf{y}}}{a} - i \ \geq \ \frac{1}{a}$.
5. Similarly, we obtain: $i + 1 - \frac{B_{\mathbf{y}}}{b} \ \geq \ \frac{1}{b}$.

Pugh's omega test (1/2)

Lemma 39 (William Pugh)

If we have:

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \ \geq \ (a-1)(b-1) \tag{6.12}$$

then $F(\mathbf{y})$ holds.

Proof.

1. Consider the closed interval: $I \ := \ \left(\frac{A_{\mathbf{y}}}{a}, \frac{B_{\mathbf{y}}}{b}\right)$.
2. If I does not contain an integer, then we have:
$$i < \frac{A_{\mathbf{y}}}{a} \leq \frac{B_{\mathbf{y}}}{b} < i+1, \quad \text{where} \quad i \ = \ \left\lfloor \frac{A_{\mathbf{y}}}{a} \right\rfloor. \tag{6.13}$$
3. Let $\rho := \mathrm{rem}(A_{\mathbf{y}}, a)$. Since $i < \frac{A_{\mathbf{y}}}{a}$ holds, we have:
$$A_{\mathbf{y}} \ = \ i\,a + \rho \ \text{ and } 0 < \rho < a, \tag{6.14}$$
4. from which we deduce: $\frac{A_{\mathbf{y}}}{a} - i \ \geq \ \frac{1}{a}$.
5. Similarly, we obtain: $i + 1 - \frac{B_{\mathbf{y}}}{b} \ \geq \ \frac{1}{b}$.
6. From the above two inequalities, elementary manipulations yield:
$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \ \leq \ ab - a - b. \tag{6.15}$$

Pugh's omega test (1/2)

Lemma 39 (William Pugh)

If we have:

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \;\geq\; (a-1)(b-1) \qquad (6.12)$$

then $F(\mathbf{y})$ *holds.*

Proof.

1. Consider the closed interval: $I \;:=\; \left(\frac{A_{\mathbf{y}}}{a}, \frac{B_{\mathbf{y}}}{b}\right)$.
2. If I does not contain an integer, then we have:
$$i < \frac{A_{\mathbf{y}}}{a} \leq \frac{B_{\mathbf{y}}}{b} < i+1, \quad \text{where} \quad i \;=\; \left\lfloor \frac{A_{\mathbf{y}}}{a} \right\rfloor. \qquad (6.13)$$
3. Let $\rho := \mathrm{rem}(A_{\mathbf{y}}, a)$. Since $i < \frac{A_{\mathbf{y}}}{a}$ holds, we have:
$$A_{\mathbf{y}} \;=\; i\,a + \rho \;\; \text{and} \; 0 < \rho < a, \qquad (6.14)$$
4. from which we deduce: $\frac{A_{\mathbf{y}}}{a} - i \;\geq\; \frac{1}{a}$.
5. Similarly, we obtain: $i + 1 - \frac{B_{\mathbf{y}}}{b} \;\geq\; \frac{1}{b}$.
6. From the above two inequalities, elementary manipulations yield:
$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \;\leq\; ab - a - b. \qquad (6.15)$$
7. Therefore, if the above inequality does not hold, that is, if $aB_{\mathbf{y}} - bA_{\mathbf{y}} \;\geq\; (a-1)(b-1)$ does hold, then I contains an integer.

Pugh's omega test (2/2)

Theorem 40

Define $\kappa(a,b) := \lceil \frac{(a-1)(b-1)}{a'} \rceil$. *Then, Formula* $F(\mathbf{y})$ *is equivalent to:*

$$((a-1)(b-1) \leq aB_{\mathbf{y}} - bA_{\mathbf{y}}) \bigvee_{k=\kappa(a,b)}^{k=b-1} (\mathbf{y} \in E_k) \wedge (a'k \leq a'B_{\mathbf{y}} - b'A_{\mathbf{y}}) \,. \tag{6.16}$$

Proof.

This is a direct consequence of William Pugh's lemma and Harris Williams' projection formula □

Remark 14

1. *William Pugh's lemma reduces significantly the number of "cuts"*
2. *To take a concrete example, say with* $a = 7$ *and* $b = 11$*:*
   - a. *with Williams' projection alone* k *ranges from* 0 *to* 10*,*
   - b. *with William Pugh's lemma,* k *ranges from* 8 *to* 10*.*

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection$(f_1, \ldots, f_s; x)$ computing $D(\mathbf{y})$.

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection$(f_1, \ldots, f_s; x)$ computing $D(\mathbf{y})$.

1. If $f_1, \ldots, f_s$ only count lower (resp. upper) bounds for x, then return true.

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection$(f_1, \ldots, f_s; x)$ computing $D(\mathbf{y})$.

1. If $f_1, \ldots, f_s$ only count lower (resp. upper) bounds for x, then return true.
2. Initialize $D(\mathbf{y})$ to true.

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection($f_1, \ldots, f_s; x$) computing $D(\mathbf{y})$.

1. If $f_1, \ldots, f_s$ only count lower (resp. upper) bounds for x, then return true.
2. Initialize $D(\mathbf{y})$ to true.
3. For each pair (A, B) consisting of a lower bound and an upper bound of x, replace $D(\mathbf{y})$ with $D(\mathbf{y}) \wedge$ Projection(A, B), where Projection(A, B) is given by Pugh's omega test.

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection$(f_1, \ldots, f_s; x)$ computing $D(\mathbf{y})$.

1. If $f_1, \ldots, f_s$ only count lower (resp. upper) bounds for x, then return true.
2. Initialize $D(\mathbf{y})$ to true.
3. For each pair (A, B) consisting of a lower bound and an upper bound of x, replace $D(\mathbf{y})$ with $D(\mathbf{y}) \wedge \text{Projection}(A, B)$, where Projection$(A, B)$ is given by Pugh's omega test.
4. Convert $D(\mathbf{y})$ to DNF yielding a formula of the form
$$S_0 \vee (C_1 \wedge S_1) \vee \cdots \vee (C_e \wedge S_e) \tag{6.17}$$
where

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection$(f_1, \ldots, f_s; x)$ computing $D(\mathbf{y})$.

1. If $f_1, \ldots, f_s$ only count lower (resp. upper) bounds for x, then return true.
2. Initialize $D(\mathbf{y})$ to true.
3. For each pair (A, B) consisting of a lower bound and an upper bound of x, replace $D(\mathbf{y})$ with $D(\mathbf{y}) \wedge \text{Projection}(A, B)$, where Projection$(A, B)$ is given by Pugh's omega test.
4. Convert $D(\mathbf{y})$ to DNF yielding a formula of the form
$$S_0 \vee (C_1 \wedge S_1) \vee \cdots \vee (C_e \wedge S_e) \tag{6.17}$$
where
    a. $S_0, S_1, \ldots, S_e$ are systems of non-strict linear inequalities in the variables $\mathbf{y}$, and

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection$(f_1, \ldots, f_s; x)$ computing $D(\mathbf{y})$.

1. If $f_1, \ldots, f_s$ only count lower (resp. upper) bounds for x, then return true.
2. Initialize $D(\mathbf{y})$ to true.
3. For each pair (A, B) consisting of a lower bound and an upper bound of x, replace $D(\mathbf{y})$ with $D(\mathbf{y}) \wedge$ Projection(A, B), where Projection(A, B) is given by Pugh's omega test.
4. Convert $D(\mathbf{y})$ to DNF yielding a formula of the form
$$S_0 \vee (C_1 \wedge S_1) \vee \cdots \vee (C_e \wedge S_e) \tag{6.17}$$
where
    - a. $S_0, S_1, \ldots, S_e$ are systems of non-strict linear inequalities in the variables $\mathbf{y}$, and
    - b. $C_1, \ldots, C_e$ are systems of congruences, for B ranging through the upper bounds of x.

Integer projection: $n = 1$, s inequalities

We now describe a procedure Projection$(f_1, \ldots, f_s; x)$ computing $D(\mathbf{y})$.

1. If $f_1, \ldots, f_s$ only count lower (resp. upper) bounds for x, then return true.
2. Initialize $D(\mathbf{y})$ to true.
3. For each pair (A, B) consisting of a lower bound and an upper bound of x, replace $D(\mathbf{y})$ with $D(\mathbf{y}) \wedge$ Projection(A, B), where Projection(A, B) is given by Pugh's omega test.
4. Convert $D(\mathbf{y})$ to DNF yielding a formula of the form
$$S_0 \vee (C_1 \wedge S_1) \vee \cdots \vee (C_e \wedge S_e) \tag{6.17}$$
where
   a. $S_0, S_1, \ldots, S_e$ are systems of non-strict linear inequalities in the variables $\mathbf{y}$, and
   b. $C_1, \ldots, C_e$ are systems of congruences, for B ranging through the upper bounds of x.
   c. S_0 is the conjunction of the $((a-1)(b-1) \leq aB_{\mathbf{y}} - bA_{\mathbf{y}})$, for all pairs (A, B) of lower and upper bounds of x.

Plan

References

[1] A. Barvinok and J. E. Pommersheim. "An algorithmic theory of lattice points in polyhedra". In: New perspectives in algebraic combinatorics 38 (1999), pp. 91–147.

[2] A. I. Barvinok. "A Polynomial Time Algorithm for Counting Integral Points in Polyhedra When the Dimension is Fixed". In: Math. Oper. Res. 19.4 (1994), pp. 769–779.

[3] M. Brion. "Points entiers dans les polyedres convexes". In: Annales scientifiques de l'École normale supérieure. Vol. 21. 4. 1988, pp. 653–663.

[4] D. C. Cooper. "Theorem proving in arithmetic without multiplication". In: Machine intelligence 7.91-99 (1972), p. 300.

[5] K. Fukuda. The CDD and CDDplus Homepage. https://www.inf.ethz.ch/personal/fukudak/cdd_home/.

[6] B. Grünbaum. Convex Polytops. New York, NY, USA: Springer, 2003.

[7] C. Haase. "A survival guide to Presburger arithmetic". In: ACM SIGLOG News 5.3 (2018), pp. 67–82.

[8] R. J. Jing and M. Moreno Maza. “Computing the Integer Points of a Polyhedron, I: Algorithm”. In: Proceedings of CASC. 2017, pp. 225–241.

[9] R. J. Jing, M. Moreno-Maza, and D. Talaashrafi. “Complexity estimates for Fourier-Motzkin elimination”. In: Proceedings of CASC. Springer. 2020, pp. 282–306.

[10] R. Jing, Y. Lei, C. F. S. Maligec, and M. Moreno Maza. “Counting the Integer Points of Parametric Polytopes: A Maple Implementation”. In: Computer Algebra in Scientific Computing - 26th International Worksh Ed. by F. Boulier, C. Mou, T. M. Sadykov, and E. V. Vorozhtsov. Vol. 14938. Lecture Notes in Computer Science. Springer, 2024, pp. 140–160. URL: https://doi.org/10.1007/978-3-031-69070-9%5C_9.

[11] R. Jing and M. Moreno Maza. “Computing the Integer Points of a Polyhedron, I: Algorithm”. In: CASC 2017, Proceedings. Vol. 10490. LNCS. Springer, 2017, pp. 225–241.

[12] N. Karmarkar. "A new polynomial-time algorithm for linear programming". In: Proceedings of the sixteenth annual ACM symposium on Theory of cor STOC '84. New York, NY, USA: ACM, 1984, pp. 302–311. ISBN: 0-89791-133-4. DOI: 10.1145/800057.808695. URL: http://doi.acm.org/10.1145/800057.808695.

[13] V. Loechner. PolyLib: A library for manipulating parameterized polyhedra. 1999.

[14] J. A. D. Loera, R. Hemmecke, J. Tauzer, and R. Yoshida. "Effective lattice point counting in rational convex polytopes". In: J. Symb. Comput. 38.4 (2004), pp. 1273–1302.

[15] M. Moreno Maza and L. Wang. "Computing the Integer Hull of Convex Polyhedral Sets". In: CASC 2022, Proceedings. Ed. by F. Boulier, M. England, T. M. Sadykov, and E. V. Vorozhtsov. Vol. 13366. Lecture Notes in Computer Science. Springer, 2022, pp. 246–267.

[16] W. Pugh. "A Practical Algorithm for Exact Array Dependence Analysis". In: Commun. ACM 35.8 (1992), pp. 102–114.

[17] W. W. Pugh. "The Omega test: a fast and practical integer programming algorithm for dependence analysis". In: Proceedings Supercomputing '91, Albuquerque, NM, USA, November 1
ACM, 1991, pp. 4–13.

[18] A. Schrijver. Theory of linear and integer programming. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 1999.

[19] R. Seghir, V. Loechner, and B. Meister. "Integer affine transformations of parametric $\mathbb{Z}$-polytopes and applications to loop nest optimization". In: ACM Trans. Archit. Code Optim. 9.2 (2012), 8:1–8:27.

[20] R. Stansifer. Presburger's article on integer arithmetic: Remarks and translation. Tech. rep. Cornell University, 1984.

[21] A. Storjohann. "Algorithms for matrix canonical forms". PhD thesis. Swiss Federal Institute of Technology Zurich, 2000.

[22] S. Verdoolaege, R. Seghir, K. Beyls, V. Loechner, and M. Bruynooghe. “Counting Integer Points in Parametric Polytopes Using Barvinok’s Rational Functions”. In: Algorithmica 48.1 (2007), pp. 37–66.

[23] H. P. Williams. “Fourier-Motzkin elimination extension to integer programming problems”. In: Journal of combinatorial theory, series A 21.1 (1976), pp. 118–123.

[24] H. Williams and J. Hooker. “Integer programming as projection”. In: Discrete Optimization 22 (2016), pp. 291–311. ISSN: 1572-5286.