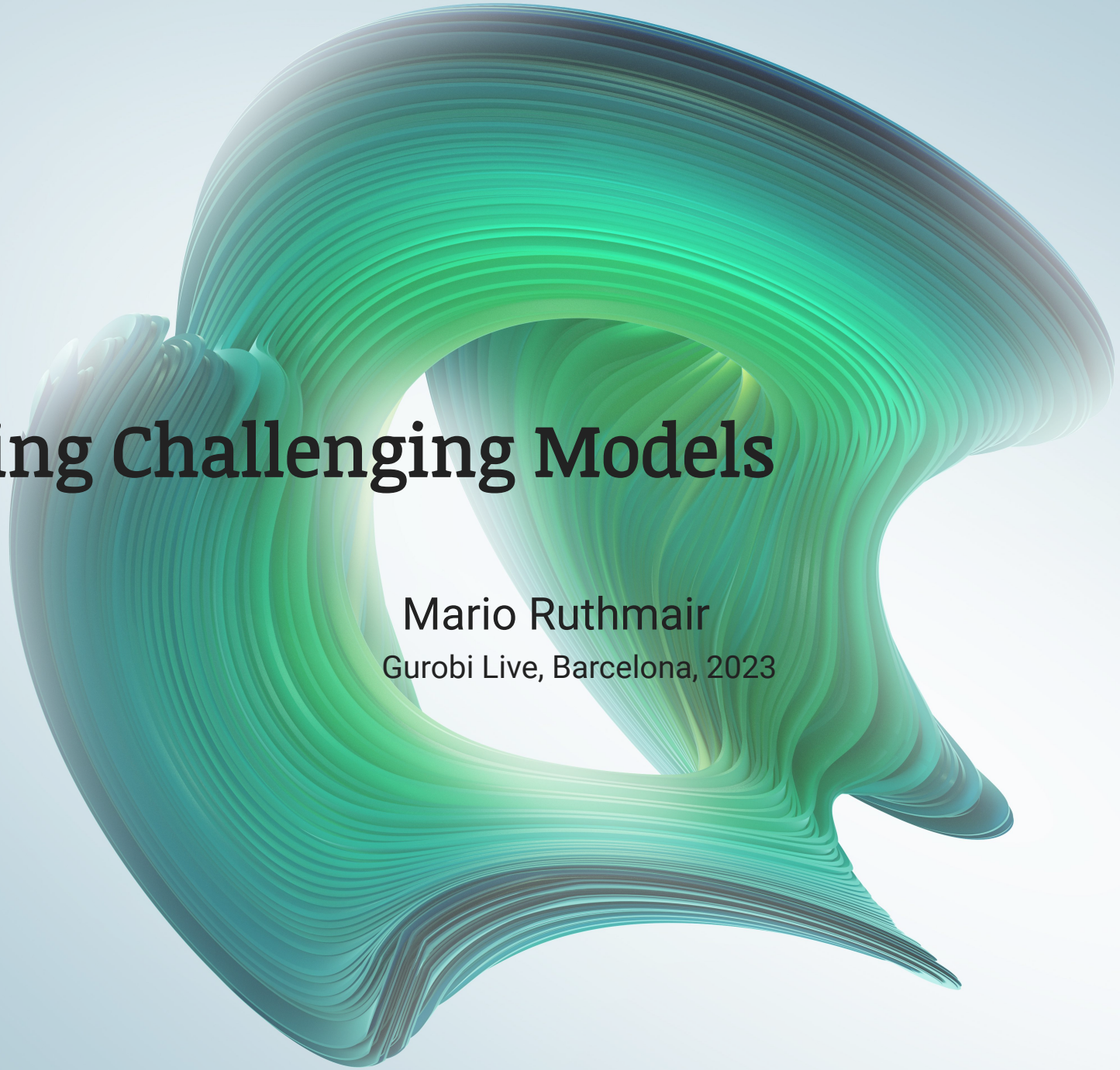


Handling Challenging Models

Mario Ruthmair

Gurobi Live, Barcelona, 2023

GUROBI LiVE!

Agenda

1. What makes a model challenging?
2. How to identify sources of difficulty?
3. What can we do?

What makes a model challenging?

What makes a model challenging?

- Returns infeasible

What makes a model challenging?

- Returns infeasible
- Takes long to solve because:

What makes a model challenging?

- Returns infeasible
- Takes long to solve because:
  - Hard to find any or good feasible solutions

What makes a model challenging?

- Returns infeasible
- Takes long to solve because:
  - Hard to find any or good feasible solutions
  - Hard to improve dual bound

What makes a model challenging?

- Returns infeasible
- Takes long to solve because:
  - Hard to find any or good feasible solutions
  - Hard to improve dual bound
  - Model is large

What makes a model challenging?

- Returns infeasible
- Takes long to solve because:
  - Hard to find any or good feasible solutions
  - Hard to improve dual bound
  - Model is large
  - Numerical issues → presentation Mastering Numerical Challenges

Infeasible Model

```
Optimize a model with 177 rows, 548 columns and 1714 nonzeros
Model fingerprint: 0x0ff2e72c
Variable types: 0 continuous, 548 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+04]
  Objective range  [5e+00, 1e+04]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+04]
Presolve removed 12 rows and 2 columns
Presolve time: 0.00s

Explored 0 nodes (0 simplex iterations) in 0.00 seconds (0.00 work unit
Thread count was 1 (of 8 available processors)

Solution count 0

Model is infeasible
Best objective -, best bound -, gap
```

Infeasible Model

```
Optimize a model with 177 rows, 548 columns and 1714 nonzeros
Model fingerprint: 0x0ff2e72c
Variable types: 0 continuous, 548 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+04]
  Objective range  [5e+00, 1e+04]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+04]
Presolve removed 12 rows and 2 columns
Presolve time: 0.00s

Explored 0 nodes (0 simplex iterations) in 0.00 seconds (0.00 work unit
Thread count was 1 (of 8 available processors)

Solution count 0

Model is infeasible
Best objective -, best bound -, gap -
```

Why infeasible?
Irreducible Inconsistent Subsystem

Why infeasible?
Irreducible Inconsistent Subsystem

- Compute an Irreducible Inconsistent Subsystem (IIS) with `computeIIS()`

Why infeasible?
Irreducible Inconsistent Subsystem

- Compute an Irreducible Inconsistent Subsystem (IIS) with `computeIIS()`
- IIS = “minimal” subset of constraints and variable bounds that conflict

Why infeasible?
Irreducible Inconsistent Subsystem

- Compute an Irreducible Inconsistent Subsystem (IIS) with `computeIIS()`
- IIS = “minimal” subset of constraints and variable bounds that conflict
- Useful to find logic errors in model building

Why infeasible?
Irreducible Inconsistent Subsystem

- Compute an Irreducible Inconsistent Subsystem (IIS) with `computeIIS()`
- IIS = “minimal” subset of constraints and variable bounds that conflict
- Useful to find logic errors in model building
- Practical limitations:

Why infeasible?
Irreducible Inconsistent Subsystem

- Compute an Irreducible Inconsistent Subsystem (IIS) with `computeIIS()`
- IIS = “minimal” subset of constraints and variable bounds that conflict
- Useful to find logic errors in model building
- Practical limitations:
  - Computational time can be huge, especially for MIPs

Why infeasible?
Irreducible Inconsistent Subsystem

- Compute an Irreducible Inconsistent Subsystem (IIS) with `computeIIS()`
- IIS = “minimal” subset of constraints and variable bounds that conflict
- Useful to find logic errors in model building
- Practical limitations:
  - Computational time can be huge, especially for MIPs
  - IIS can be large and difficult to interpret

Why infeasible?

```
Computing Irreducible Inconsistent Subsystem (IIS)...

            Constraints             |              Bounds             |  Runtime
      Min         Max       Guess   |   Min         Max       Guess   |
-------------------------------------------------------------------------------
        0         177           -   |     0        1096           -   |     0s
        2           2           2   |     2           2           2   |     0s

IIS computed: 2 constraints, 2 bounds
IIS runtime: 0.01 seconds (0.00 work units)
```

Why infeasible?

```
Computing Irreducible Inconsistent Subsystem (IIS)...

            Constraints             |              Bounds             |  Runtime
      Min        Max      Guess     |   Min        Max      Guess     |
-------------------------------------------------------------------------------
        0        177          -           0       1096          -          0s
        2          2          2           2          2          2          0s

IIS computed: 2 constraints, 2 bounds
IIS runtime: 0.01 seconds (0.00 work units)
```

IIS in LP file format (default bounds $x \geq 0$):

```
Minimize

Subject To
 c79: 93 x1 - 189 x2 + 10 x3 >= 12
 c156: x1 - x2 + x3 <= 0
Bounds
 x2 free
Generals
 x1 x2 x3
End
```

Why infeasible?
Feasibility Relaxation

Why infeasible?
Feasibility Relaxation

- Find a solution with minimum constraint violation with `feasRelax()`

Why infeasible?
Feasibility Relaxation

- Find a solution with minimum constraint violation with `feasRelax()`
- Artificial slack variables show constraint violation

Why infeasible?
Feasibility Relaxation

- Find a solution with minimum constraint violation with `feasRelax()`
- Artificial slack variables show constraint violation
- Useful to find input data errors

Why infeasible?
Feasibility Relaxation

- Find a solution with minimum constraint violation with `feasRelax()`
- Artificial slack variables show constraint violation
- Useful to find input data errors
- Usually quite fast

Why infeasible?
Feasibility Relaxation

```
Optimize a model with 1273 rows, 1821 columns and 4083 nonzeros
Model fingerprint: 0xe02d128a
Variable types: 1273 continuous, 548 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+04]
  Objective range  [1e+00, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 1e+04]
Found heuristic solution: objective 1768.0000000
Presolve added 0 rows and 36 columns
Presolve removed 593 rows and 0 columns
Presolve time: 0.00s
Presolved: 680 rows, 1857 columns, 3096 nonzeros
Variable types: 124 continuous, 1733 integer (548 binary)
Found heuristic solution: objective 1306.0000000

Root relaxation: objective 1.269841e-01, 304 iterations, 0.00 seconds (
```

Why infeasible?
Feasibility Relaxation

```
Variable types: 124 continuous, 1733 integer (548 binary)
Found heuristic solution: objective 1306.0000000

Root relaxation: objective 1.269841e-01, 304 iterations, 0.00 seconds (

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

     0     0    0.12698    0   59 1306.00000    0.12698   100%     -
H    0     0                     1168.0000000    0.12698   100%     -
H    0     0                     1142.0000000    0.12698   100%     -
H    0     0                      866.2222222    0.12698   100%     -
H    0     0                      586.9999998    0.12698   100%     -
     0     0    1.00000    0   74  587.00000    1.00000   100%     -
H    0     0                      400.9999998    1.00000   100%     -
H    0     0                       80.0000000    1.00000  98.7%     -
     0     0    1.00000    0   72   80.00000    1.00000  98.7%     -
H    0     0                       79.0000000    1.00000  98.7%
```

Why infeasible?
Feasibility Relaxation

```
33  H    0     0                          10.0000000    1.00000  90.0%      -
34  H    0     0                           5.0000000    1.00000  80.0%      -
35       0     0    1.00000    0   16      5.00000      1.00000  80.0%      -
36       0     0    1.00000    0   16      5.00000      1.00000  80.0%      -
37       0     0    1.00000    0   16      5.00000      1.00000  80.0%      -
38       0     2    1.00000    0   16      5.00000      1.00000  80.0%      -
39  H   36    40                           4.0000000    1.00000  75.0%    3.1
40  H   79   161                           2.0000000    1.00000  50.0%    2.4
41  H  117   161                           1.0000000    1.00000  0.00%    2.2
42
43  Cutting planes:
44    Gomory: 20
45    MIR: 10
46
47  Explored 172 nodes (1178 simplex iterations) in 0.31 seconds (0.15 work
48  Thread count was 8 (of 8 available processors)
49
50  Solution count 10: 1 2 4     866 222
```

Why infeasible?
Feasibility Relaxation

Relation to IIS?

```
Minimize

Subject To
 c79: 93 x1 - 189 x2 + 10 x3 >= 12
 c156: x1 - x2 + x3 <= 0
Bounds
 x2 free
Generals
 x1 x2 x3
End
```

Why infeasible?
Feasibility Relaxation

Relation to IIS?

```
Minimize

Subject To
 c79: 93 x1 - 189 x2 + 10 x3 >= 12
 c156: x1 - x2 + x3 <= 0
Bounds
 x2 free
Generals
 x1 x2 x3
End
```

- Solution values from feasibility relaxation: `x1 = 1, x2 = 0, x3 = 0`

Why infeasible?
Feasibility Relaxation

Relation to IIS?

```
Minimize

Subject To
 c79: 93 x1 - 189 x2 + 10 x3 >= 12
 c156: x1 - x2 + x3 <= 0
Bounds
 x2 free
Generals
 x1 x2 x3
End
```

- Solution values from feasibility relaxation: `x1 = 1, x2 = 0, x3 = 0`
- Value of constraint slack variable: `ArtN_c156 = 1`

How to identify sources of difficulty?

Solver log for model `glass4`

```
Optimize a model with 396 rows, 322 columns and 1815 nonzeros
Model fingerprint: 0x18b19fdf
Variable types: 20 continuous, 302 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 8e+06]
  Objective range  [1e+00, 1e+06]
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.01s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
Optimize a model with 396 rows, 322 columns and 1815 nonzeros
Model fingerprint: 0x18b19fdf
Variable types: 20 continuous, 302 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 8e+06]
  Objective range  [1e+00, 1e+06]
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.01s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl | Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
Optimize a model with 396 rows, 322 columns and 1815 nonzeros
Model fingerprint: 0x18b19fdf
Variable types: 20 continuous, 302 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 8e+06]
  Objective range  [1e+00, 1e+06]
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.01s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
Model fingerprint: 0x18b19fdf
Variable types: 20 continuous, 302 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 8e+06]
  Objective range  [1e+00, 1e+06]
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.01s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

```

How to identify sources of difficulty?

Solver log for model `glass4`

```
  Matrix range     [1e+00, 8e+06]
  Objective range  [1e+00, 1e+06]
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.01s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

     0     0 8.0000e+08    0   72 3.1334e+09 8.0000e+08  74.5%     -
H    0     0                     2.200019e+09 8.0000e+08  63.6%     -
H    0     0                     2.000016e+09 8.0000e+08  60.0%     -
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.01s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

     0     0 8.0000e+08    0   72 3.1334e+09 8.0000e+08  74.5%     -
H    0     0                     2.200019e+09 8.0000e+08  63.6%     -
H    0     0                     2.000016e+09 8.0000e+08  60.0%     -
     0     0 8.0000e+08    0   72 2.0000e+09 8.0000e+08  60.0%     -
H    0     0                     1.930017e+09 8.0000e+08  58.5%     -
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

     0     0 8.0000e+08    0   72 3.1334e+09 8.0000e+08  74.5%     -
H    0     0                       2.200019e+09 8.0000e+08  63.6%     -
H    0     0                       2.000016e+09 8.0000e+08  60.0%     -
     0     0 8.0000e+08    0   72 2.0000e+09 8.0000e+08  60.0%     -
H    0     0                       1.930017e+09 8.0000e+08  58.5%     -
     0     0 8.0000e+08    0   72 1.9300e+09 8.0000e+08  58.5%     -
     0     0 8.0000e+08    0   83 1.9300e+09 8.0000e+08  58.5%     -
     0     0 8.0000e+08    0   78 1.9300e+09 8.0000e+08  58.5%     -
     0     2 8.0000e+08    0   78 1.9300e+09 8.0000e+08  58.5%     -
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

     0     0 8.0000e+08    0   72 3.1334e+09 8.0000e+08  74.5%     -
H    0     0                      2.200019e+09 8.0000e+08  63.6%     -
H    0     0                      2.000016e+09 8.0000e+08  60.0%     -
     0     0 8.0000e+08    0   72 2.0000e+09 8.0000e+08  60.0%     -
H    0     0                      1.930017e+09 8.0000e+08  58.5%     -
     0     0 8.0000e+08    0   72 1.9300e+09 8.0000e+08  58.5%     -
     0     0 8.0000e+08    0   83 1.9300e+09 8.0000e+08  58.5%     -
     0     0 8.0000e+08    0   78 1.9300e+09 8.0000e+08  58.5%     -
     0     2 8.0000e+08    0   78 1.9300e+09 8.0000e+08  58.5%     -
H   95   111                      1.816682e+09 8.0000e+08  56.0%   7.6
H  414   410                      1.750015e+09 8.0000e+08  54.3%   4.5
H 1334   991                      1.733348e+09 8.0000e+08  53.8%   5.5
```

How to identify sources of difficulty?

Solver log for model `glass4`

21	H	0	0				2.200019e+09	8.0000e+08	63.6%	-
22	H	0	0				2.000016e+09	8.0000e+08	60.0%	-
23		0	0	8.0000e+08	0	72	2.0000e+09	8.0000e+08	60.0%	-
24	H	0	0				1.930017e+09	8.0000e+08	58.5%	-
25		0	0	8.0000e+08	0	72	1.9300e+09	8.0000e+08	58.5%	-
26		0	0	8.0000e+08	0	83	1.9300e+09	8.0000e+08	58.5%	-
27		0	0	8.0000e+08	0	78	1.9300e+09	8.0000e+08	58.5%	-
28		0	2	8.0000e+08	0	78	1.9300e+09	8.0000e+08	58.5%	-
29	H	95	111				1.816682e+09	8.0000e+08	56.0%	7.6
30	H	414	410				1.750015e+09	8.0000e+08	54.3%	4.5
31	H	1334	991				1.733348e+09	8.0000e+08	53.8%	5.5
32	H	1367	965				1.700014e+09	8.0000e+08	52.9%	5.4
33	*	18952	13298			62	1.700014e+09	8.0000e+08	52.9%	4.3
34	*	21945	13292			48	1.700014e+09	8.0001e+08	52.9%	4.4
35	*	21946	13278			48	1.700014e+09	8.0001e+08	52.9%	4.4
36	H	23596	14591				1.700013e+09	8.0001e+08	52.9%	4.4
37	H	27260	14584				1.688903e+09	8.0001e+08	52.6%	4.3
38	H	30543	15189				1.644458e+09	8.6667e+08	47.3%	4.2

How to identify sources of difficulty?

Solver log for model `glass4`

```
30  H  414    410                          1.750015e+09 8.0000e+08  54.3%    4.5
31  H 1334    991                          1.733348e+09 8.0000e+08  53.8%    5.5
32  H 1367    965                          1.700014e+09 8.0000e+08  52.9%    5.4
33  *18952 13298              62           1.700014e+09 8.0000e+08  52.9%    4.3
34  *21945 13292              48           1.700014e+09 8.0001e+08  52.9%    4.4
35  *21946 13278              48           1.700014e+09 8.0001e+08  52.9%    4.4
36  H23596 14591                           1.700013e+09 8.0001e+08  52.9%    4.4
37  H27260 14584                           1.688903e+09 8.0001e+08  52.6%    4.3
38  H30543 15189                           1.644458e+09 8.6667e+08  47.3%    4.2
39  H30547 14430                           1.644458e+09 8.6667e+08  47.3%    4.2
40  H30550 13710                           1.644458e+09 8.6667e+08  47.3%    4.2
41  H30555 13027                           1.600013e+09 8.6667e+08  45.8%    4.2
42   31146 13383 1.0500e+09   62    42 1.6000e+09 9.0000e+08  43.8%    4.4
43   95341 33758 1.6000e+09   76    13 1.6000e+09 1.0000e+09  37.5%    6.0
44   187457 77051 1.5000e+09   60    37 1.6000e+09 1.0000e+09  37.5%    5.9
45  *263218 109630               73     1.600013e+09 1.0000e+09  37.5%    5.7
46   266480 111130 infeasible    55       1.6000e+09 1.0000e+09  37.5%    5.7
47  *270267 80393                71     1.550013e+09 1.0000e+09  35.5%    5.7
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
 50  H293401 79715                          1.500013e+09 1.0000e+09  33.3%    5.7
 51  H293417 57180                          1.475014e+09 1.0000e+09  32.2%    5.7
 52  H299029 55344                          1.460014e+09 1.0000e+09  31.5%    5.7
 53   312999 55932 1.4600e+09   52    35 1.4600e+09 1.0072e+09  31.0%    5.7
 54  *331032 52143                 75     1.400013e+09 1.1000e+09  21.4%    5.9
 55   350453 53694 infeasible      54       1.4000e+09 1.1000e+09  21.4%    6.0
 56   384990 51565 1.4000e+09      50    28 1.4000e+09 1.2000e+09  14.3%    6.2
 57   413646 58415 1.3000e+09      68    27 1.4000e+09 1.2000e+09  14.3%    6.3
 58  H413897 11468                          1.200013e+09 1.2000e+09  0.00%    6.3
 59
 60  Cutting planes:
 61    Learned: 1
 62    Gomory: 10
 63    Cover: 3
 64    Implied bound: 11
 65    Projected implied bound: 1
 66    MIR: 12
 67    Flow cover: 43
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
 384990 51565 1.4000e+09   50   28 1.4000e+09 1.2000e+09  14.3%   6.2
 413646 58415 1.3000e+09   68   27 1.4000e+09 1.2000e+09  14.3%   6.3
H413897 11468                     1.200013e+09 1.2000e+09  0.00%   6.3

Cutting planes:
  Learned: 1
  Gomory: 10
  Cover: 3
  Implied bound: 11
  Projected implied bound: 1
  MIR: 12
  Flow cover: 43
  RLT: 2
  Relax-and-lift: 10

Explored 413963 nodes (2629055 simplex iterations) in 40.32 seconds (23
Thread count was 8 (of 8 available processors)

-----------------------------------------------------------------------
```

How to identify sources of difficulty?

Solver log for model `glass4`

```
Cutting planes:
  Learned: 1
  Gomory: 10
  Cover: 3
  Implied bound: 11
  Projected implied bound: 1
  MIR: 12
  Flow cover: 43
  RLT: 2
  Relax-and-lift: 10

Explored 413963 nodes (2629055 simplex iterations) in 40.32 seconds (23
Thread count was 8 (of 8 available processors)

Solution count 10: 1.20001e+09 1.40001e+09 1.46001e+09 ... 1.60001e+09

Optimal solution found (tolerance 1.00e-04)
Best objective 1.200012600000e+09, best bound 1.200006650000e+09, gap 0
```

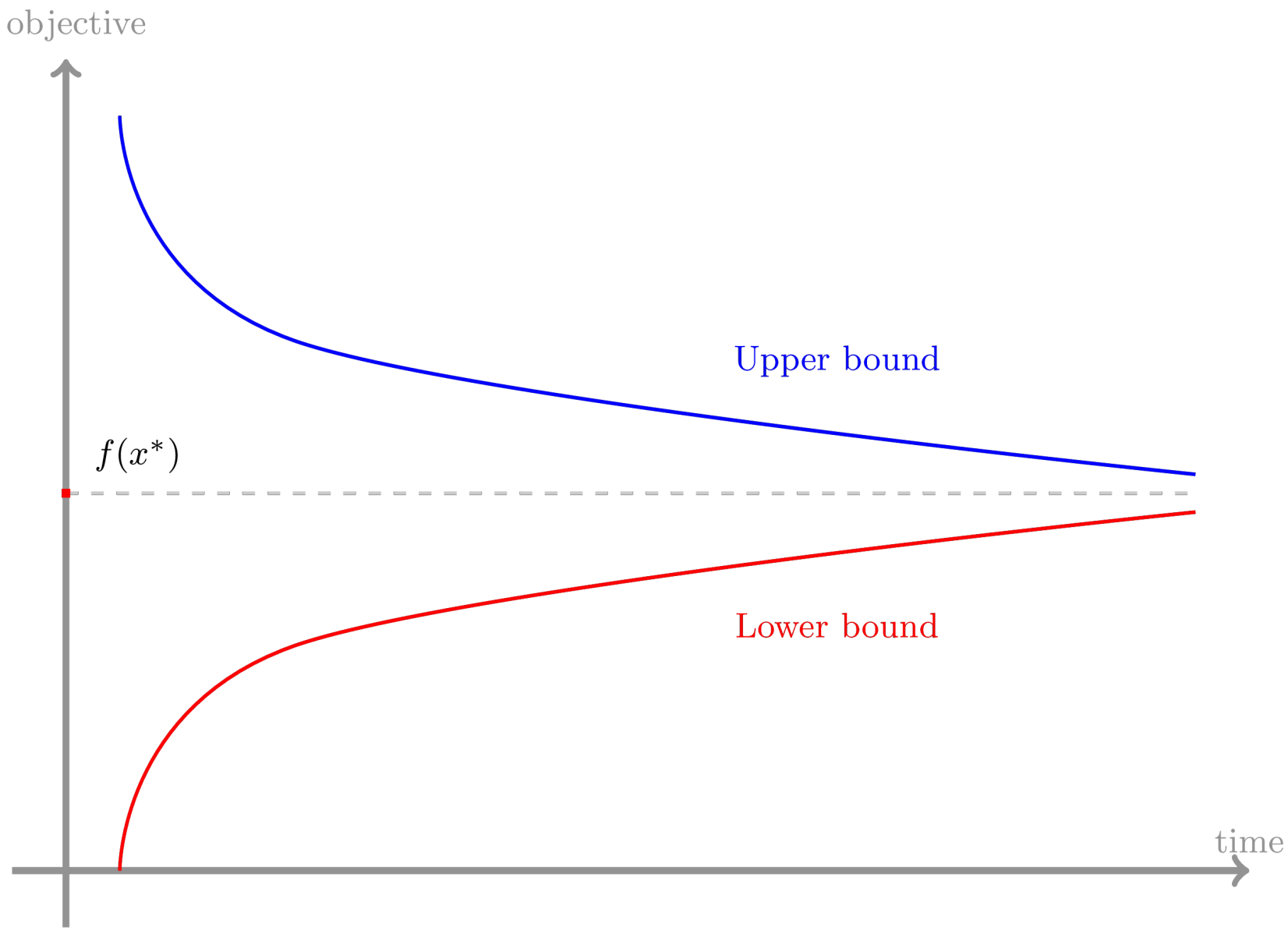
How to identify sources of difficulty?

Solver log for model `glass4`

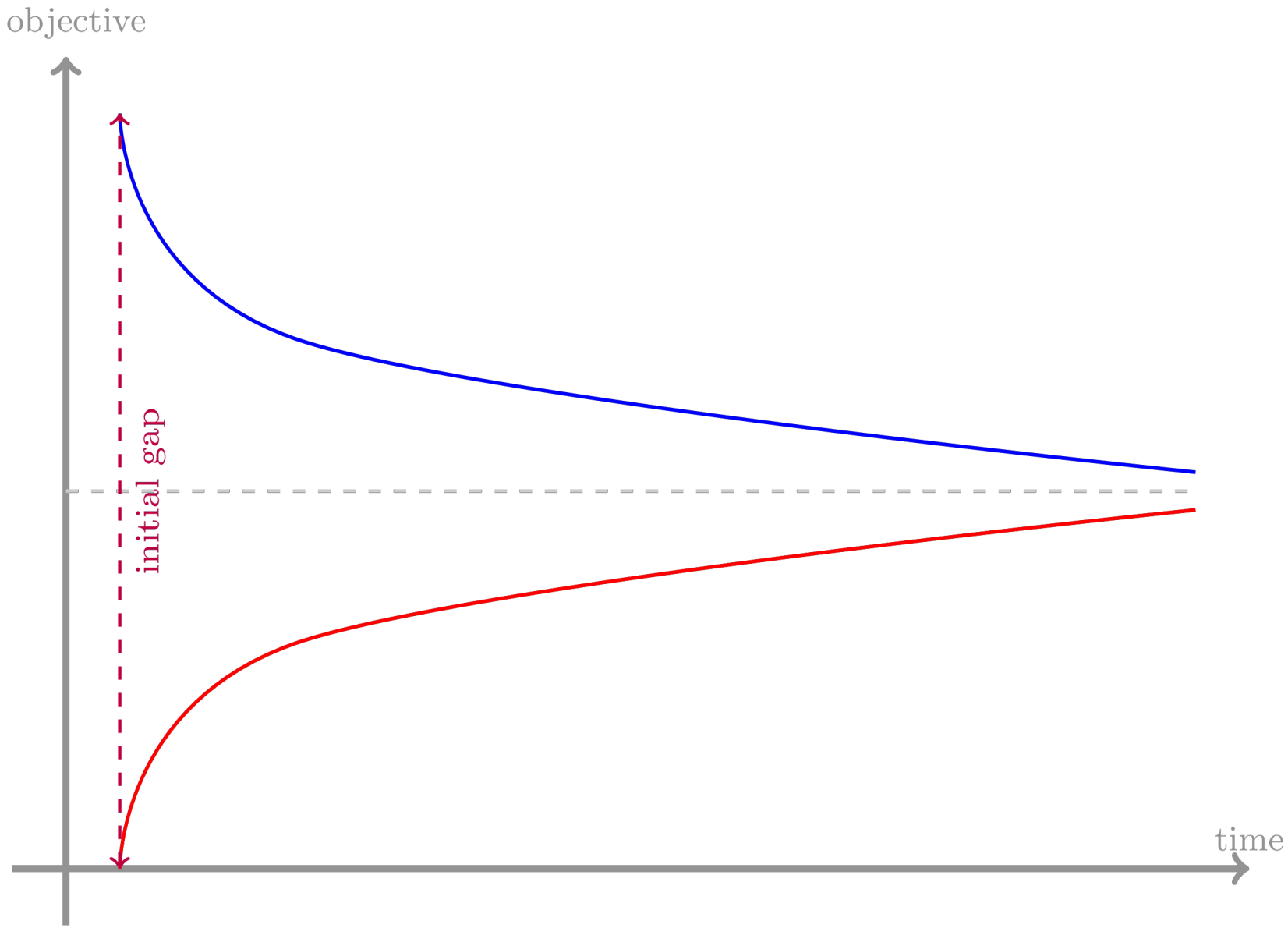
```
Cutting planes:
  Learned: 1
  Gomory: 10
  Cover: 3
  Implied bound: 11
  Projected implied bound: 1
  MIR: 12
  Flow cover: 43
  RLT: 2
  Relax-and-lift: 10

Explored 413963 nodes (2629055 simplex iterations) in 40.32 seconds (23
Thread count was 8 (of 8 available processors)

Solution count 10: 1.20001e+09 1.40001e+09 1.46001e+09 ... 1.60001e+09

Optimal solution found (tolerance 1.00e-04)
Best objective 1.200012600000e+09, best bound 1.200006650000e+09, gap 0
```

Solution progress

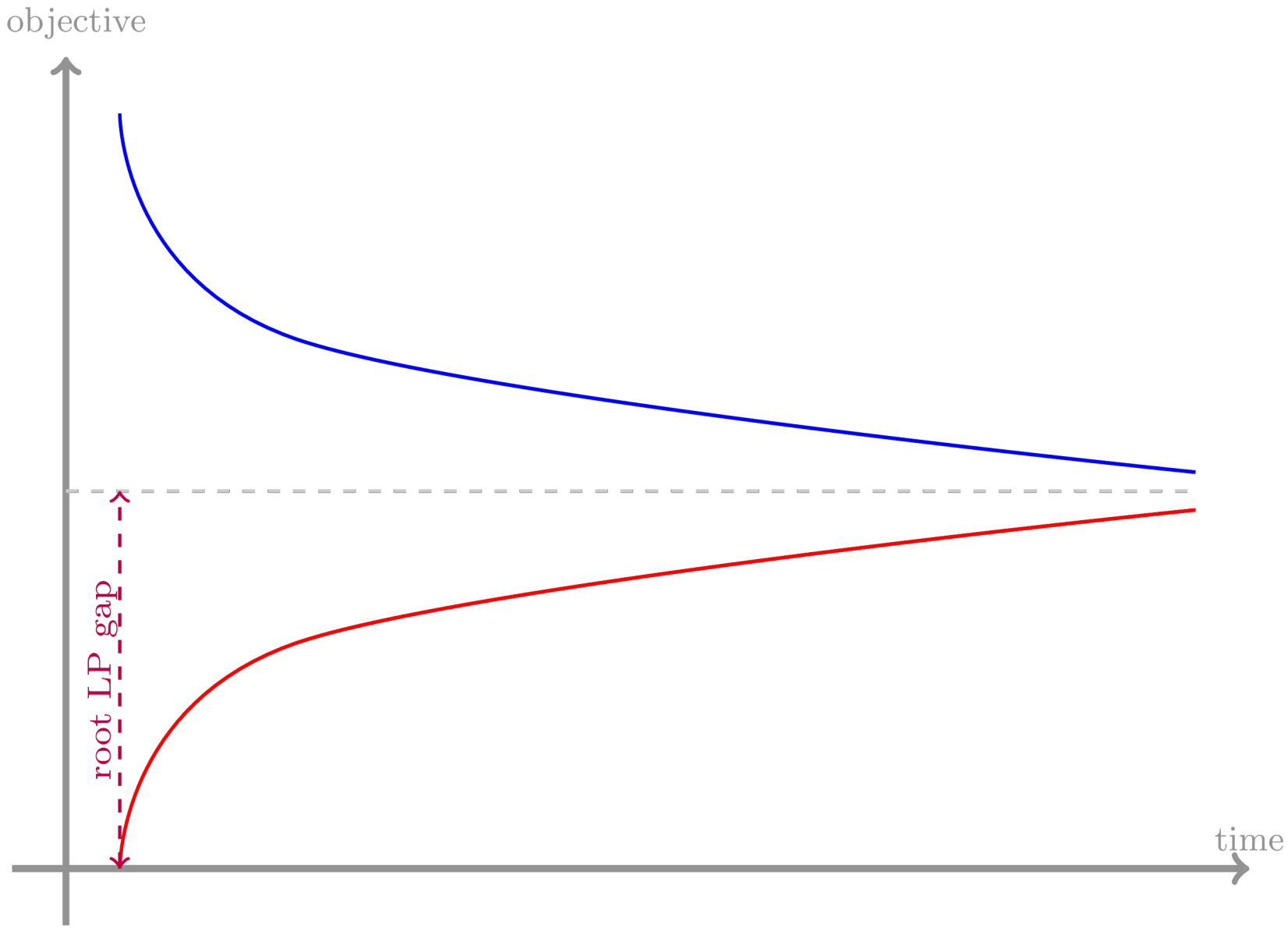
Solution progress



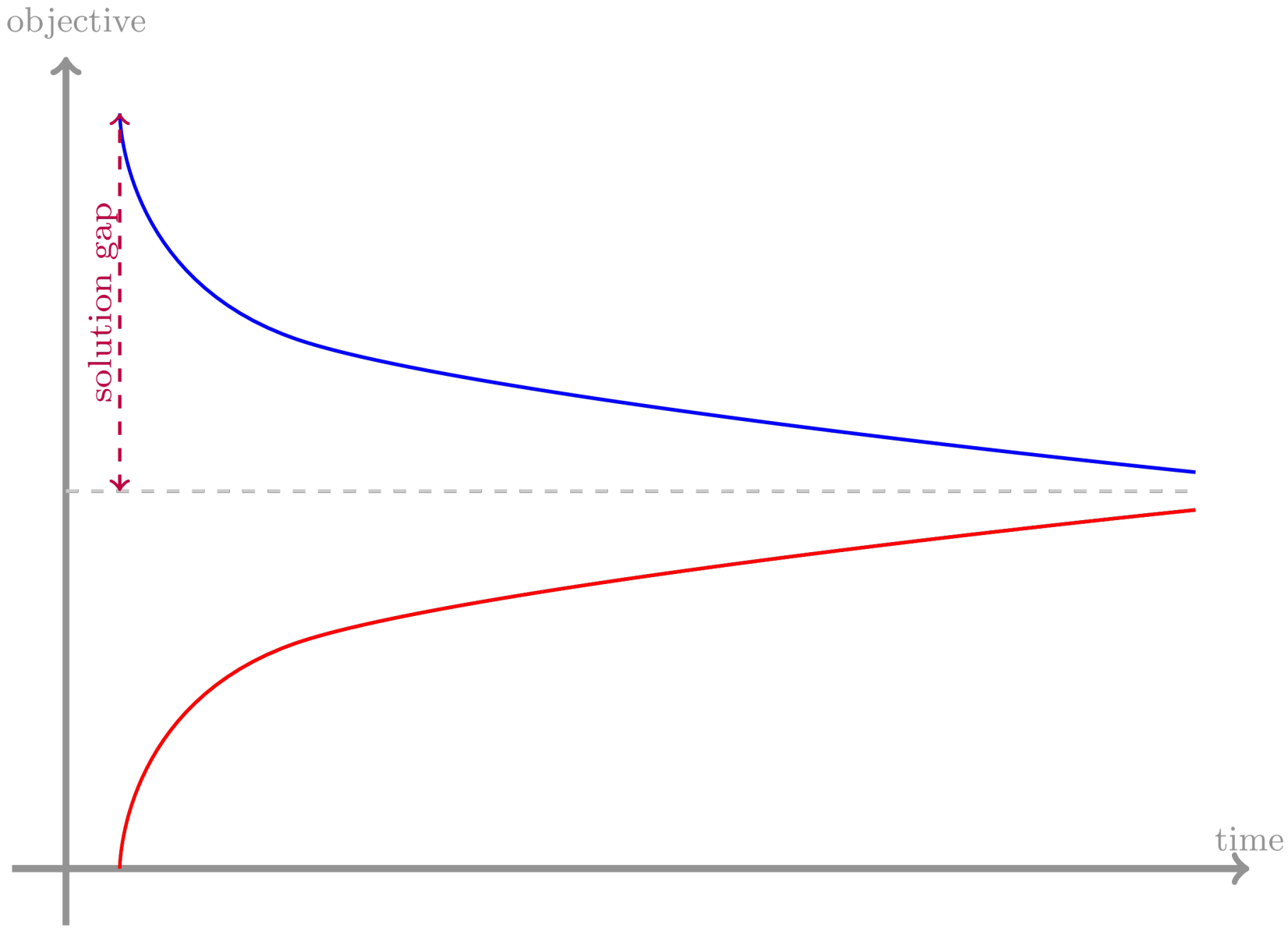
Solution progress



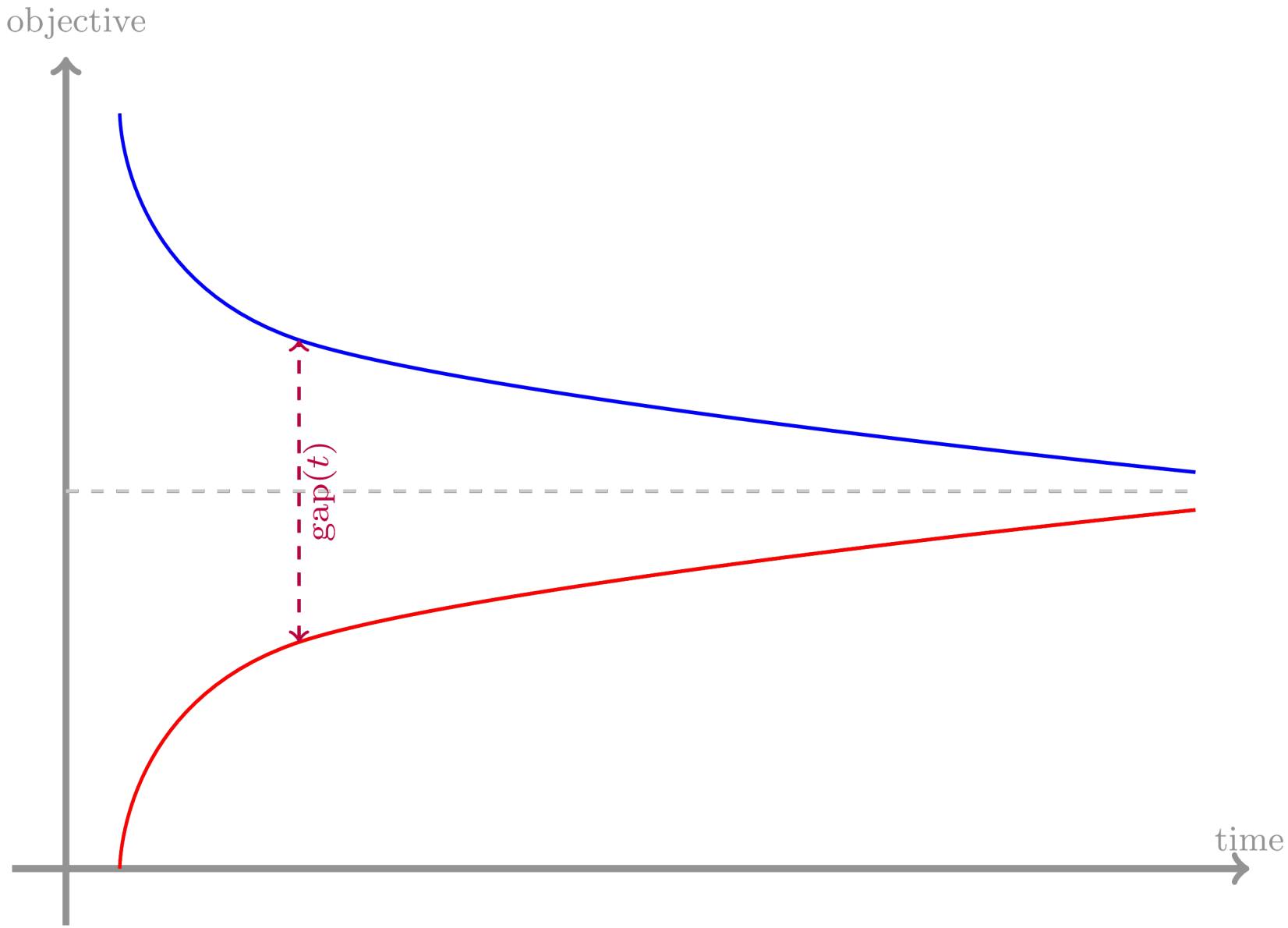
Solution progress



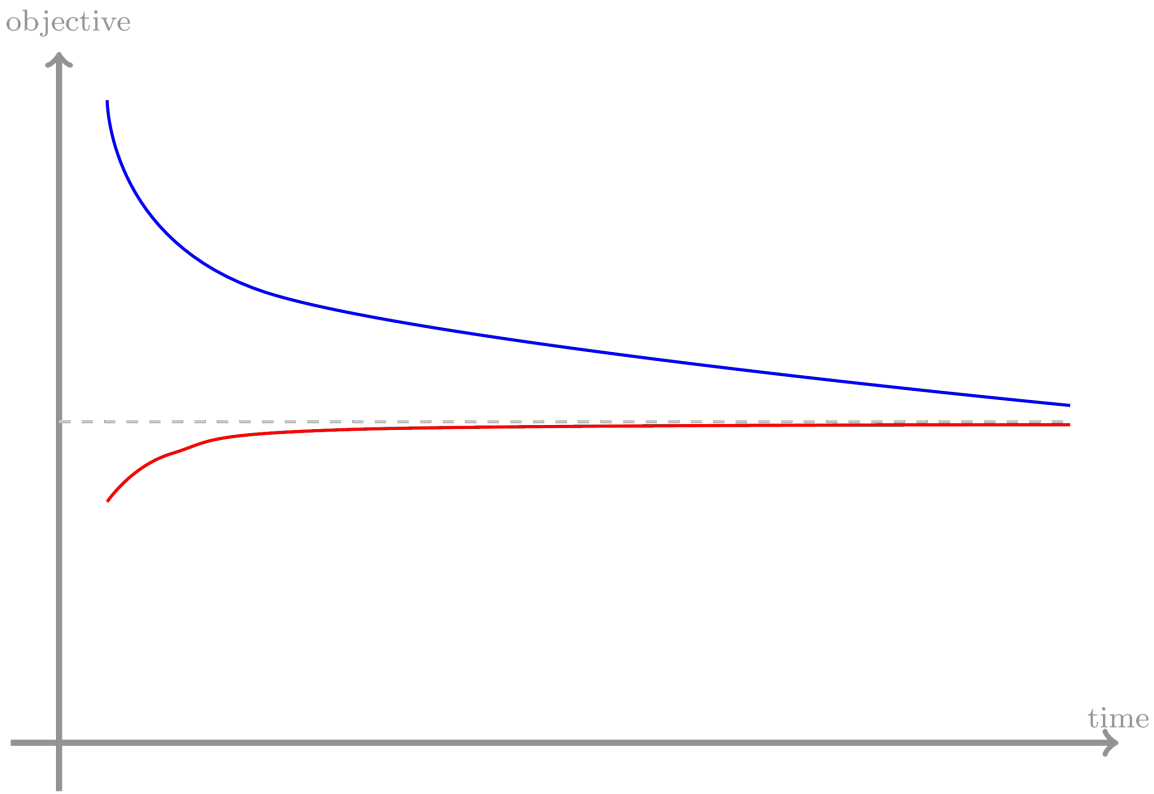
Solution progress



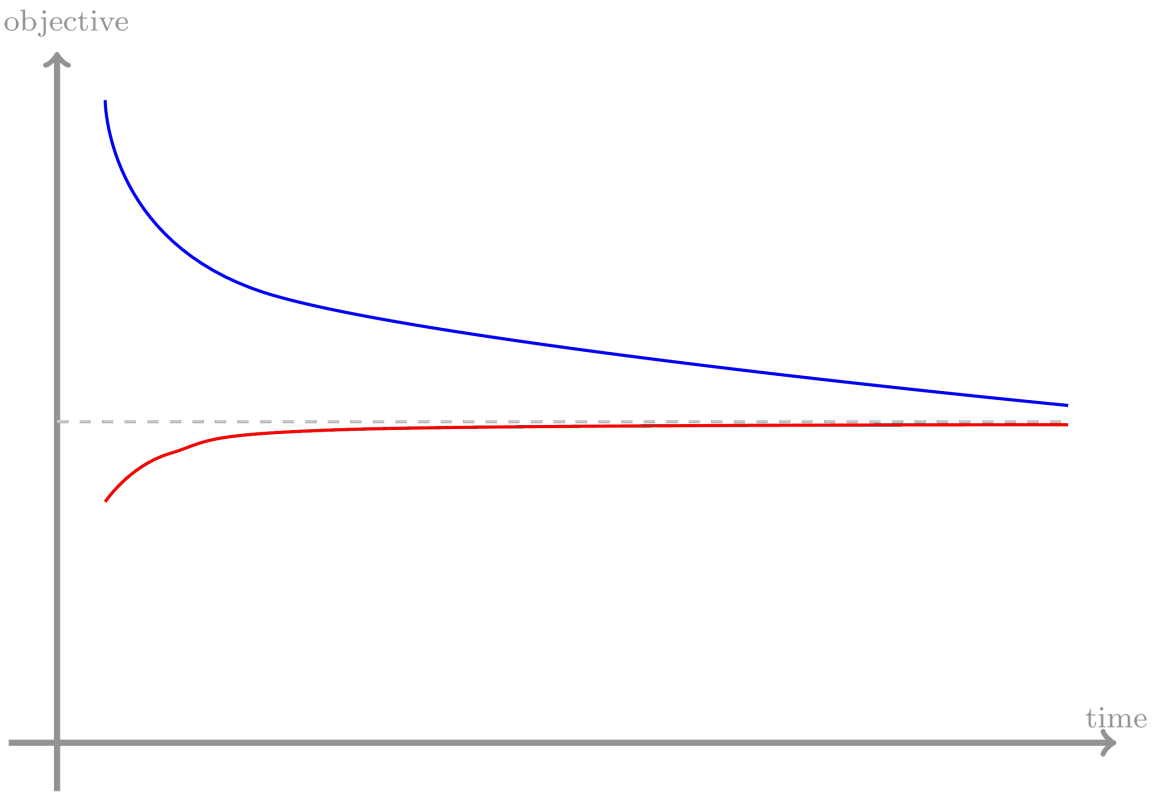
Solution progress



Bad solution, good bound

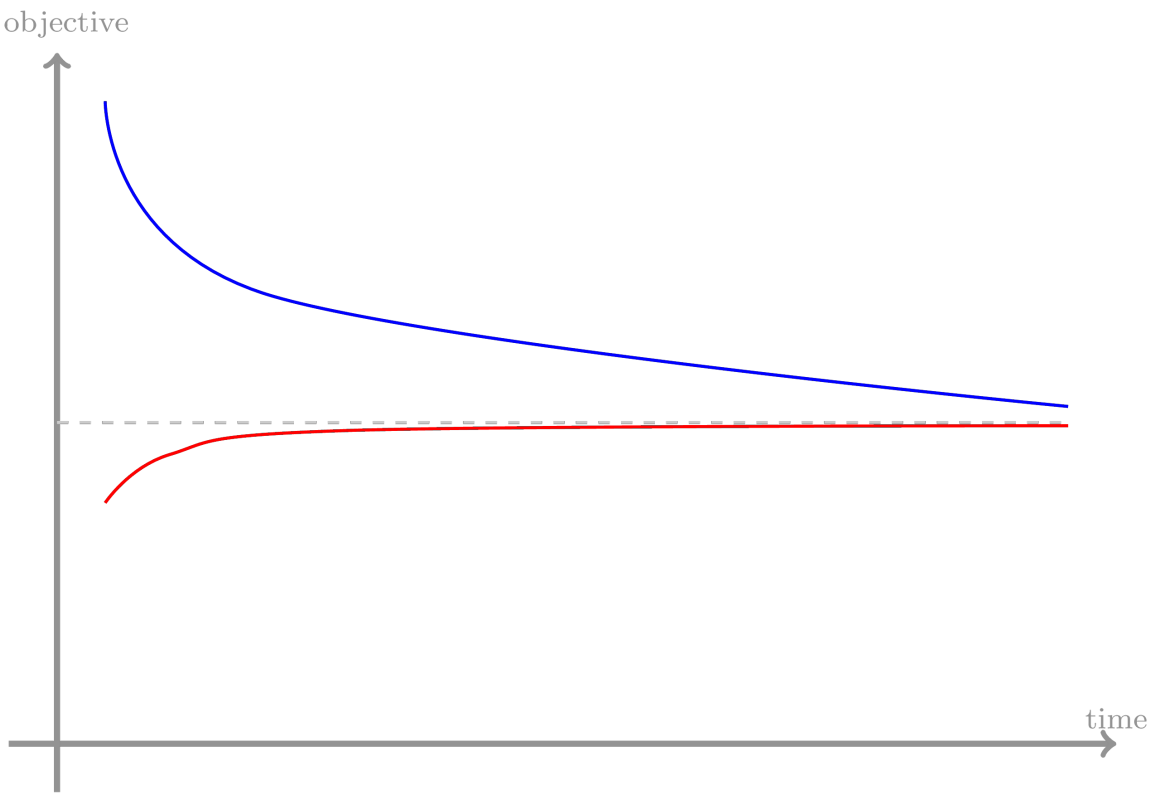


Bad solution, good bound



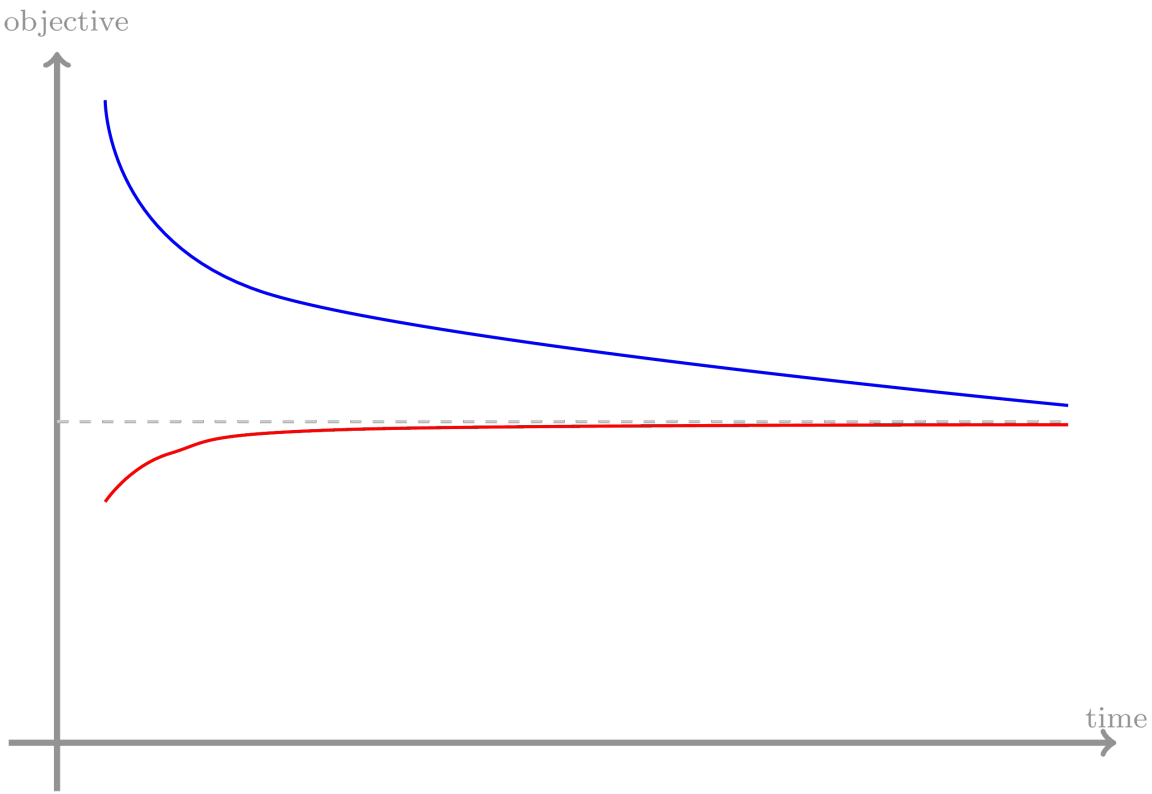
- Bound is strong and converges quickly

Bad solution, good bound



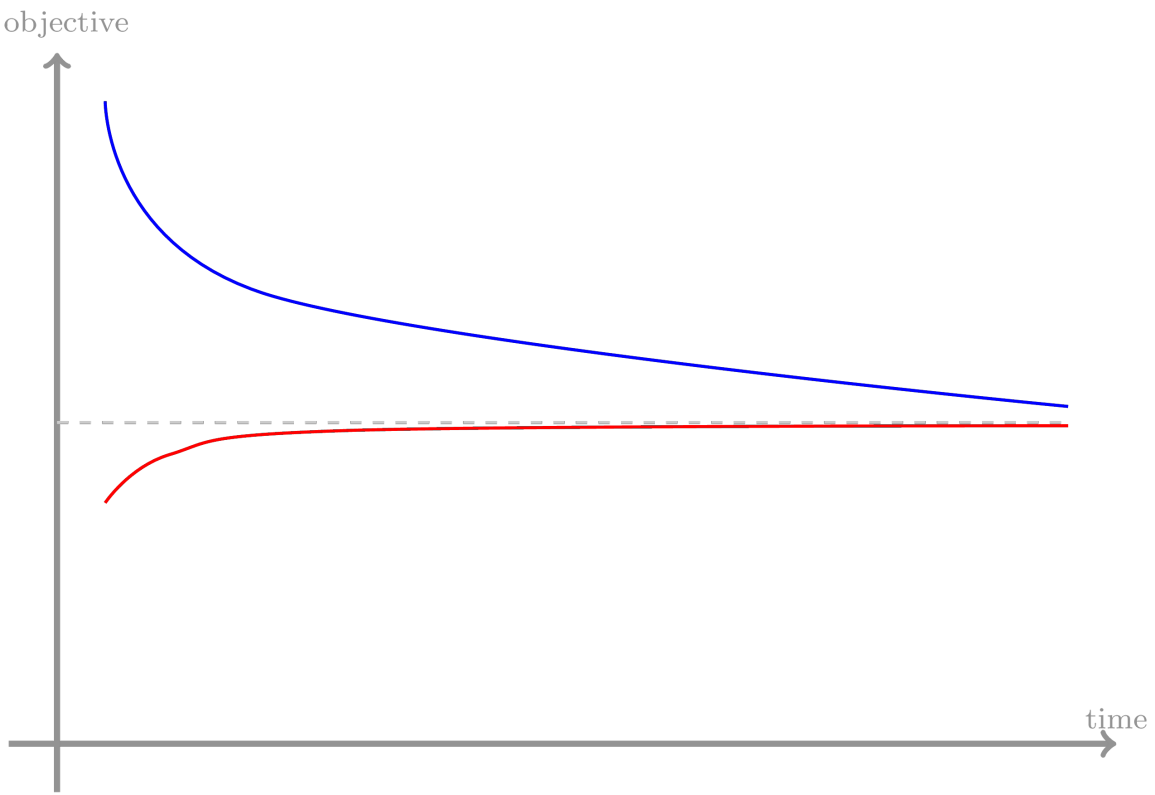
- Bound is strong and converges quickly
- Solutions improve slowly →

Bad solution, good bound



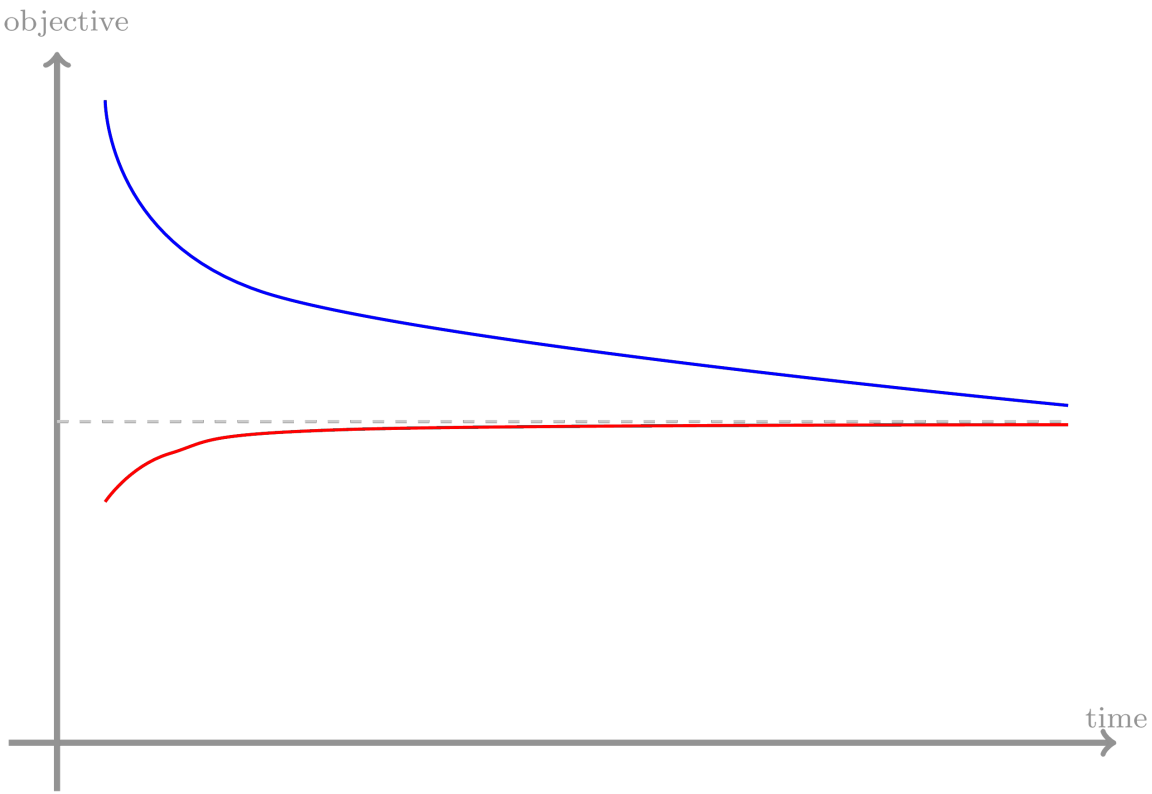
- Bound is strong and converges quickly
- Solutions improve slowly →
  - Solver parameters focusing on heuristics

Bad solution, good bound



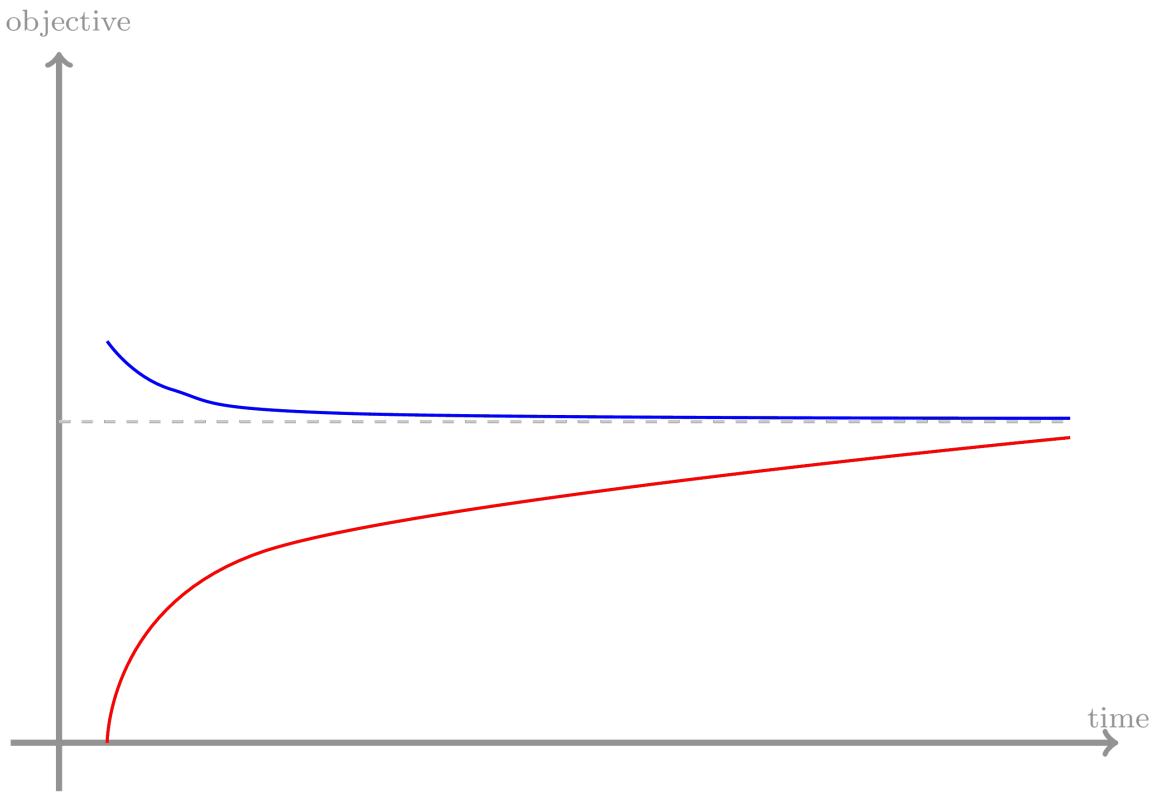
- Bound is strong and converges quickly
- Solutions improve slowly →
  - Solver parameters focusing on heuristics
  - Enlarge feasible region

Bad solution, good bound

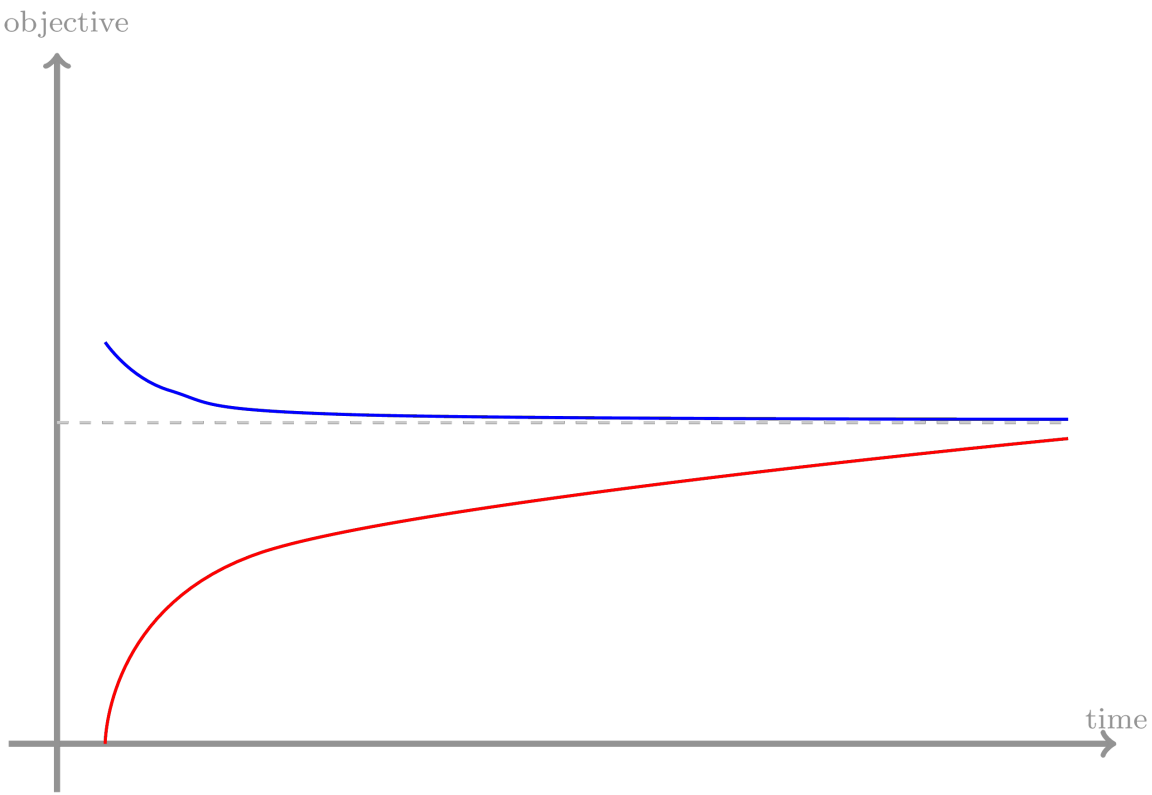


- Bound is strong and converges quickly
- Solutions improve slowly →
  - Solver parameters focusing on heuristics
  - Enlarge feasible region
  - Custom heuristics

Good solution, bad bound

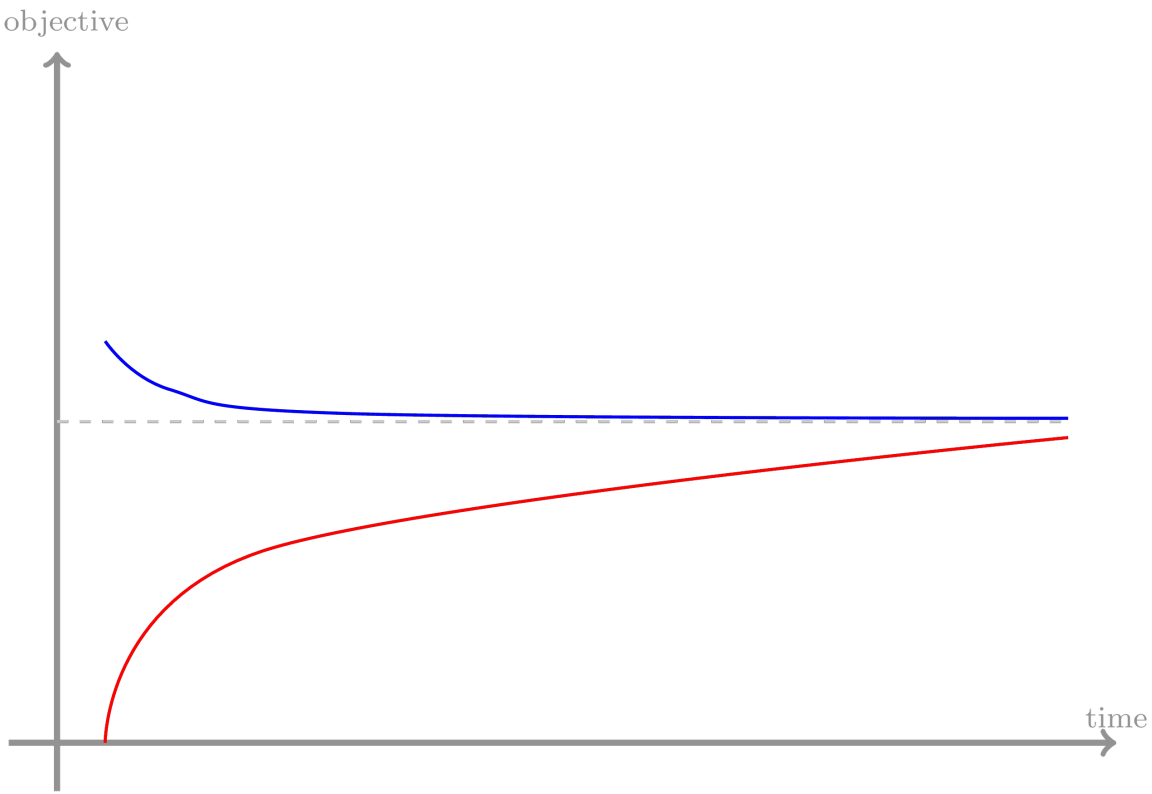


Good solution, bad bound



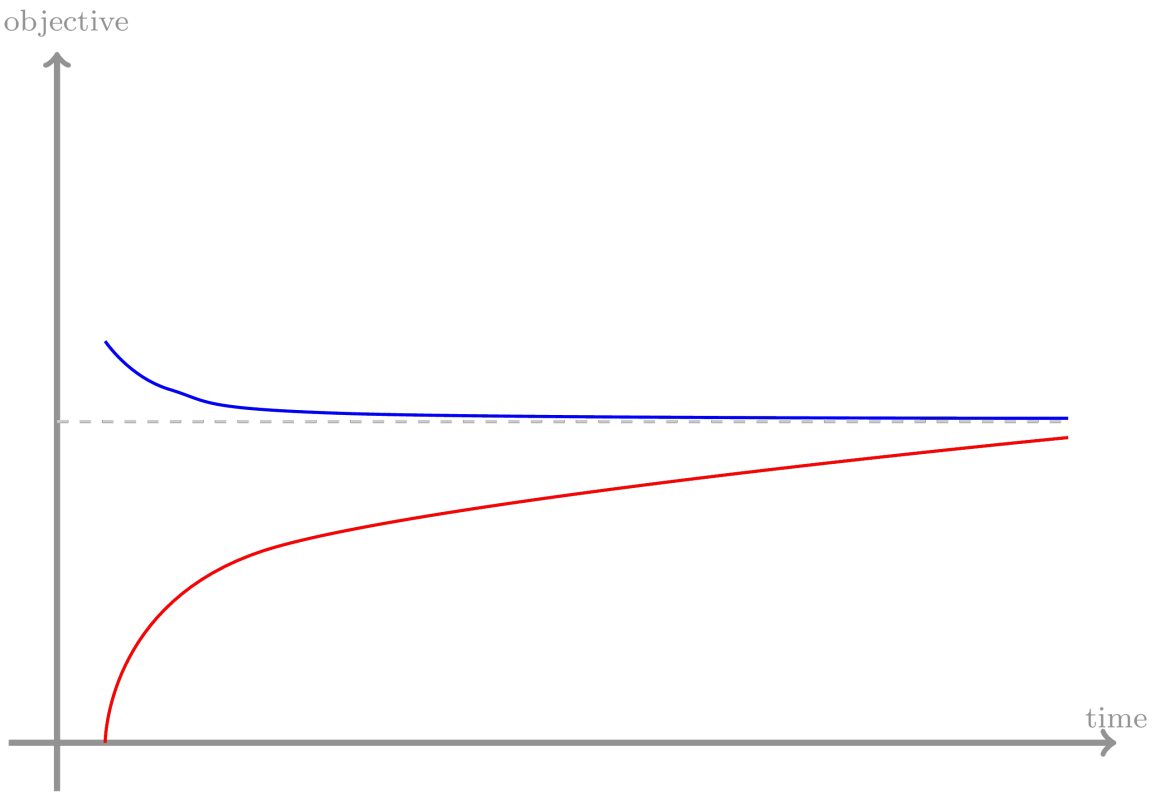
- Solution improves quickly due to internal heuristics (or user solutions)

Good solution, bad bound



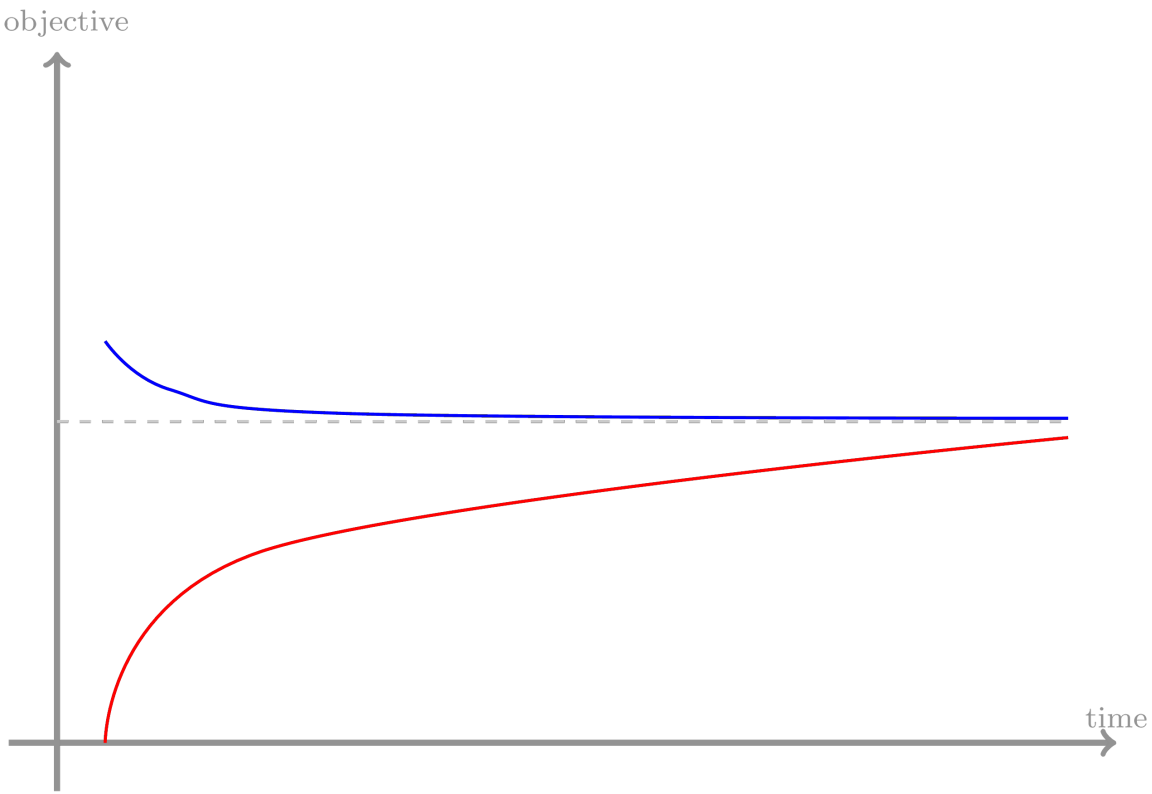
- Solution improves quickly due to internal heuristics (or user solutions)
- Bound is weak and increases slowly →

Good solution, bad bound



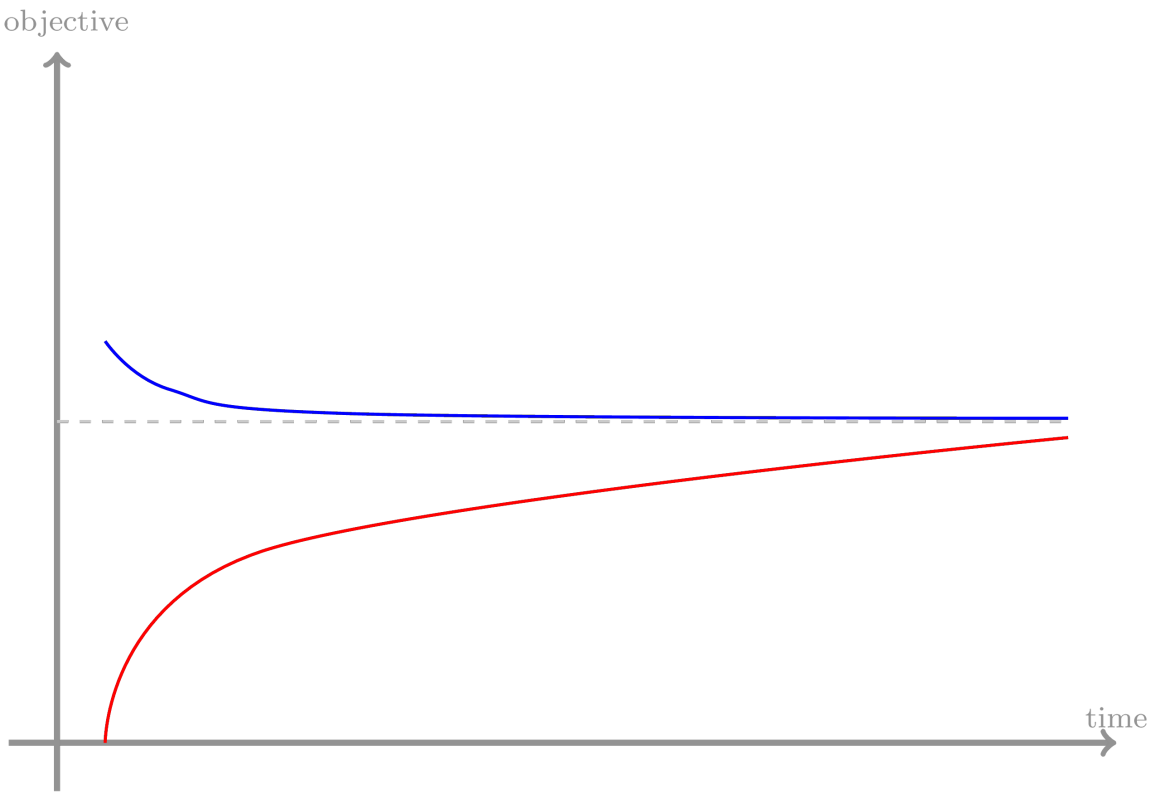
- Solution improves quickly due to internal heuristics (or user solutions)
- Bound is weak and increases slowly →
  - Solver parameters focusing on bound

Good solution, bad bound



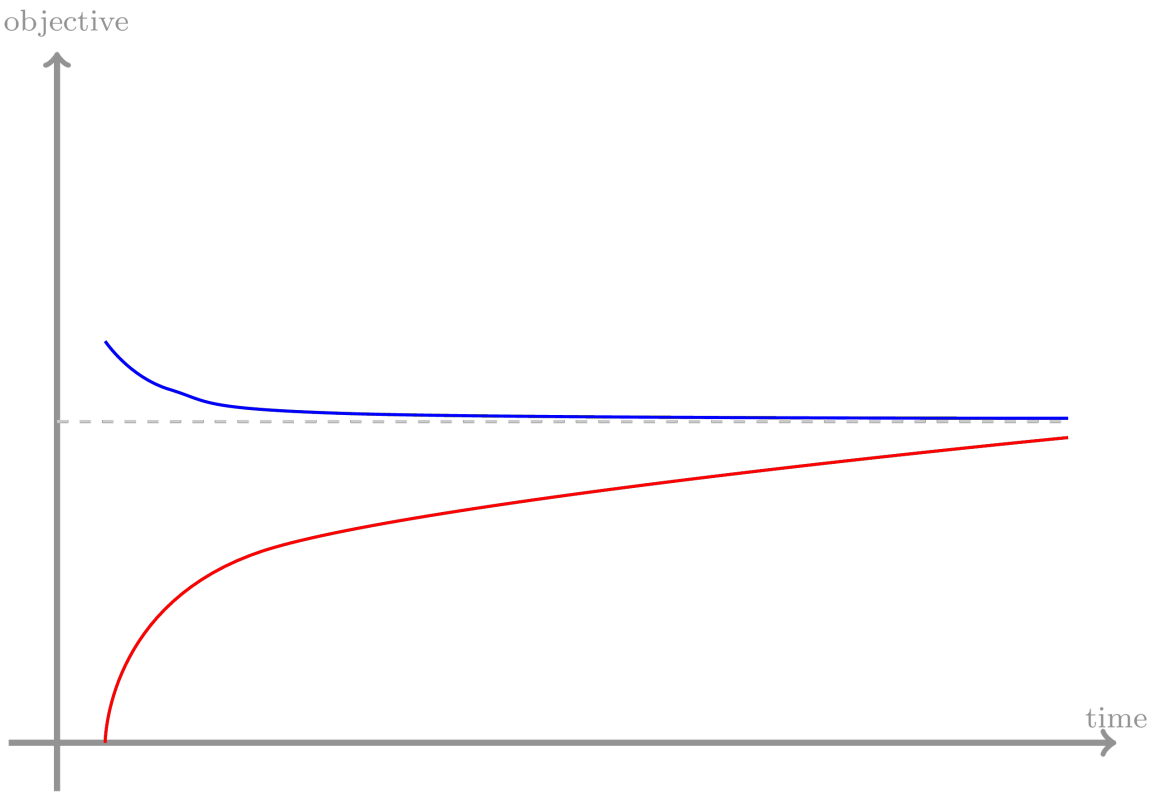
- Solution improves quickly due to internal heuristics (or user solutions)
- Bound is weak and increases slowly →
  - Solver parameters focusing on bound
  - More cuts

Good solution, bad bound



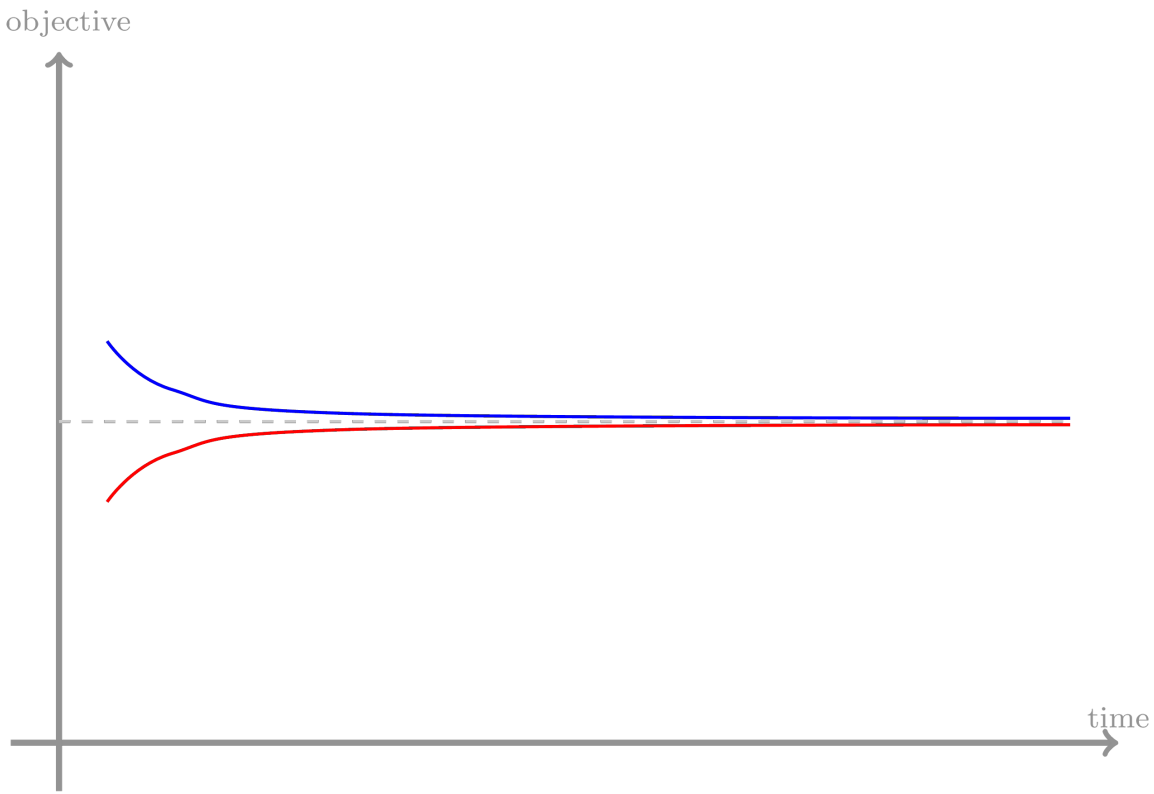
- Solution improves quickly due to internal heuristics (or user solutions)
- Bound is weak and increases slowly →
  - Solver parameters focusing on bound
  - More cuts
  - More preprocessing

Good solution, bad bound

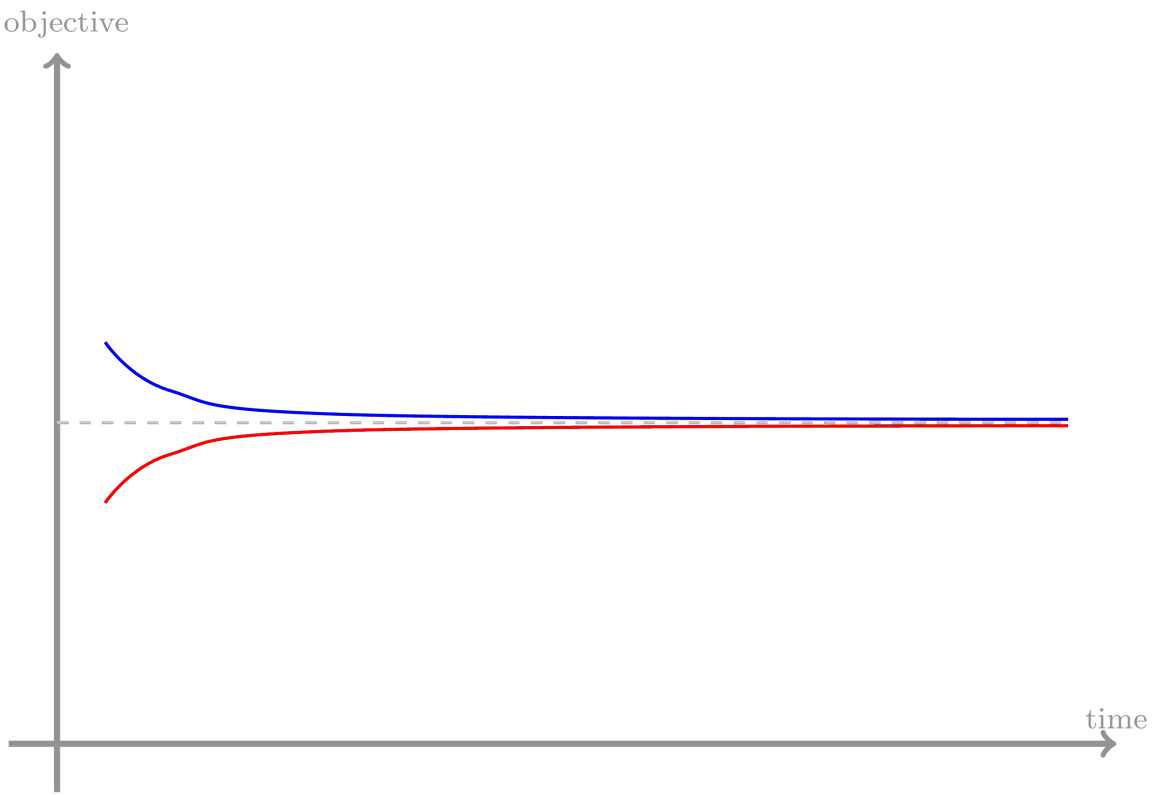


- Solution improves quickly due to internal heuristics (or user solutions)
- Bound is weak and increases slowly →
  - Solver parameters focusing on bound
  - More cuts
  - More preprocessing
  - Reformulate model

Tailing off

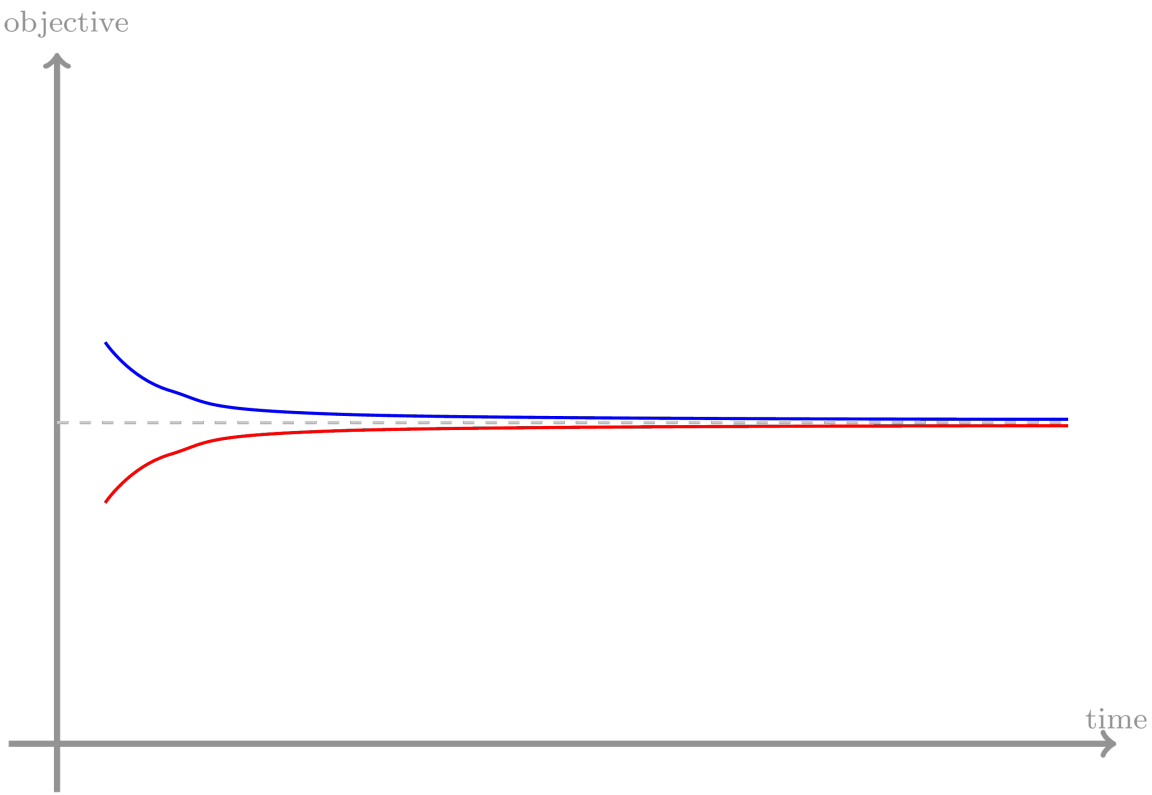


Tailing off



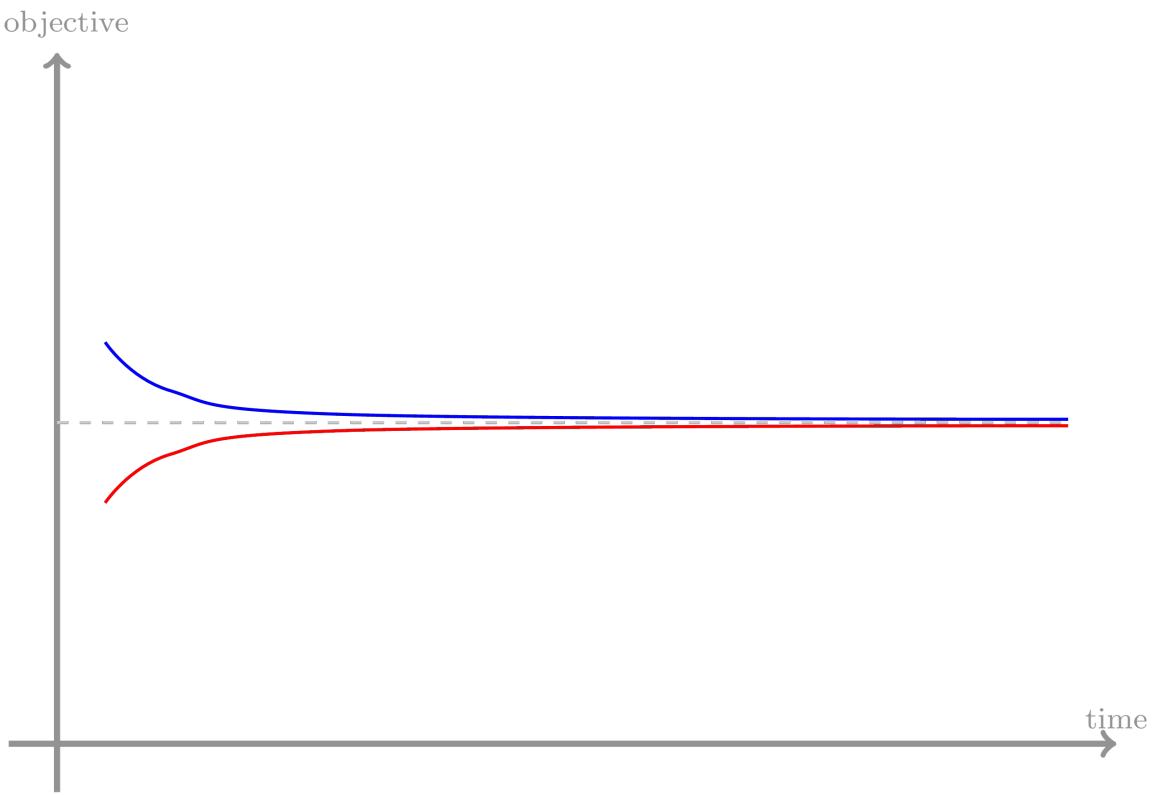
- Solution and bound improve quickly but take long to close the gap

Tailing off



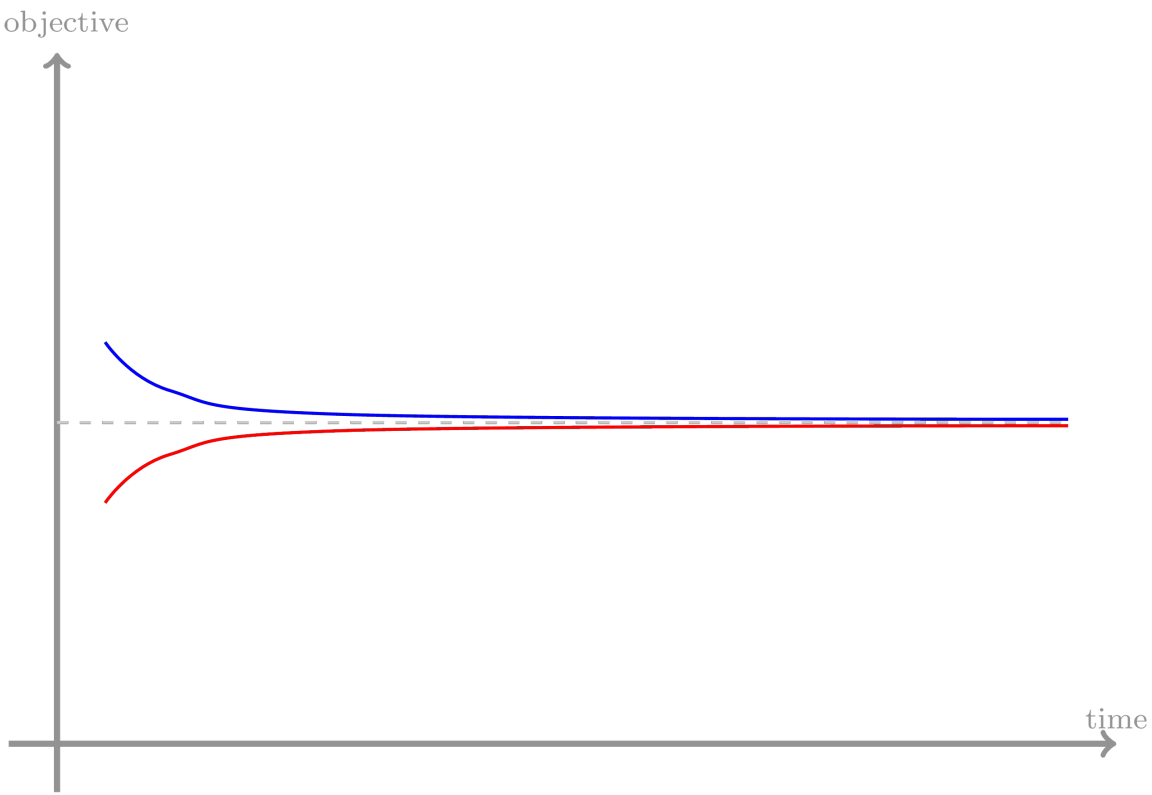
- Solution and bound improve quickly but take long to close the gap
- Think about termination criteria:

Tailing off



- Solution and bound improve quickly but take long to close the gap
- Think about termination criteria:
  - How precise is data?

Tailing off



- Solution and bound improve quickly but take long to close the gap
- Think about termination criteria:
  - How precise is data?
  - How precise is model?

Gurobi LogTools

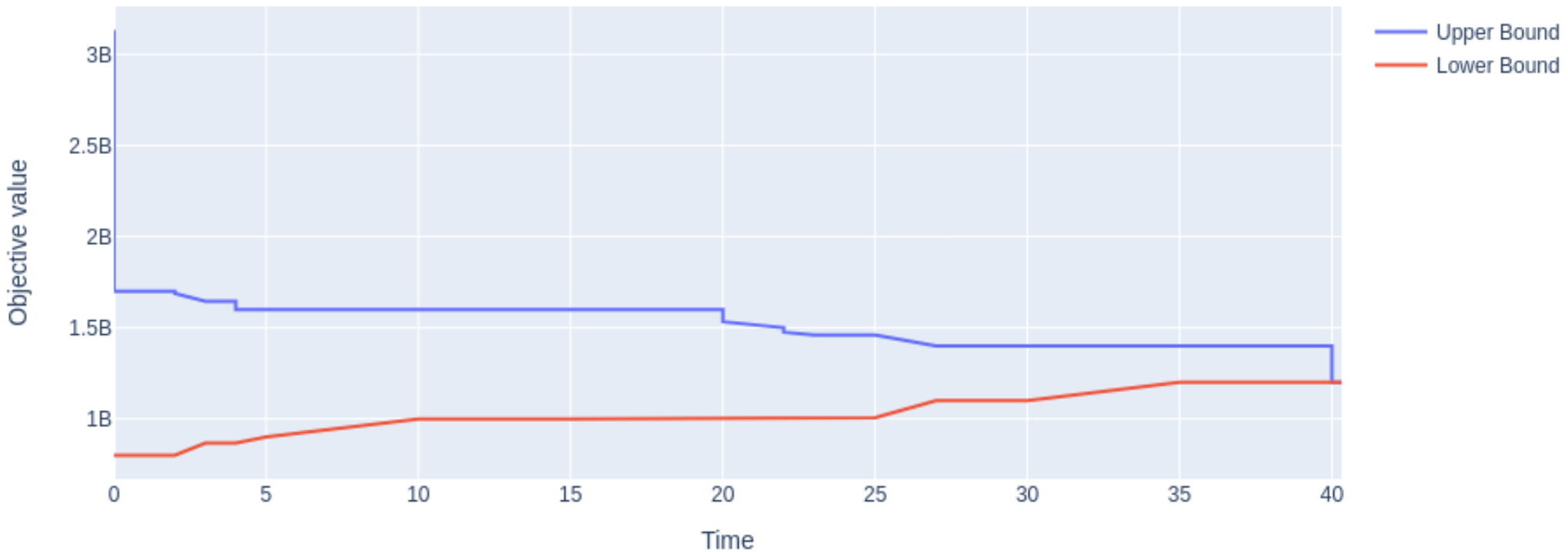
- Python package to parse and analyze solver logs
- `pip install gurobi-logtools`

Gurobi LogTools

- Python package to parse and analyze solver logs
- `pip install gurobi-logtools`

```
import gurobi_logtools as glt
import plotly.graph_objects as go

results = glt.parse(["*.log"])
log = results.progress("nodelog")

fig = go.Figure()
fig.add_trace(go.Scatter(x=log["Time"], y=log["Incumbent"], name="Upper
fig.add_trace(go.Scatter(x=log["Time"], y=log["BestBd"], name="Lower Bo
fig.update_xaxes(title_text="Time")
fig.update_yaxes(title_text="Objective value")
fig.show()
```

Gurobi LogTools
Example: `glass4`



Tune Solver Parameters

Example: glass4

```
Set parameter Heuristics to value 0.5
Set parameter Cuts to value 0
...
     Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

     0     0 8.0000e+08    0   72 3.1334e+09 8.0000e+08  74.5%     -
H    0     0                      2.200019e+09 8.0000e+08  63.6%     -
H    0     0                      2.000016e+09 8.0000e+08  60.0%     -
     0     0 8.0000e+08    0   72 2.0000e+09 8.0000e+08  60.0%     -
     0     2 8.0000e+08    0   72 2.0000e+09 8.0000e+08  60.0%     -
H  171   176                      2.000016e+09 8.0000e+08  60.0%   2.2
H  173   176                      1.966684e+09 8.0000e+08  59.3%   2.2
H  203   315                      1.966684e+09 8.0000e+08  59.3%   2.2
H  321   445                      1.900017e+09 8.0000e+08  57.9%   2.3
H  395   445                      1.855571e+09 8.0000e+08  56.9%   2.3
H  598   578                      1.800015e+09 8.0000e+08  55.6%   2.4
H  680   921                      1.794460e+09 8.0000e+08  55.4%   2.6
```

Tune Solver Parameters

Example: glass4

```
Set parameter Heuristics to value 0.5
Set parameter Cuts to value 0
...
     Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

     0     0 8.0000e+08    0   72 3.1334e+09 8.0000e+08  74.5%     -
H    0     0                       2.200019e+09 8.0000e+08  63.6%     -
H    0     0                       2.000016e+09 8.0000e+08  60.0%     -
     0     0 8.0000e+08    0   72 2.0000e+09 8.0000e+08  60.0%     -
     0     2 8.0000e+08    0   72 2.0000e+09 8.0000e+08  60.0%     -
H  171   176                       2.000016e+09 8.0000e+08  60.0%   2.2
H  173   176                       1.966684e+09 8.0000e+08  59.3%   2.2
H  203   315                       1.966684e+09 8.0000e+08  59.3%   2.2
H  321   445                       1.900017e+09 8.0000e+08  57.9%   2.3
H  395   445                       1.855571e+09 8.0000e+08  56.9%   2.3
H  598   578                       1.800015e+09 8.0000e+08  55.6%   2.4
H  680   921                       1.794460e+09 8.0000e+08  55.4%   2.6
```

Tune Solver Parameters

Example: glass4

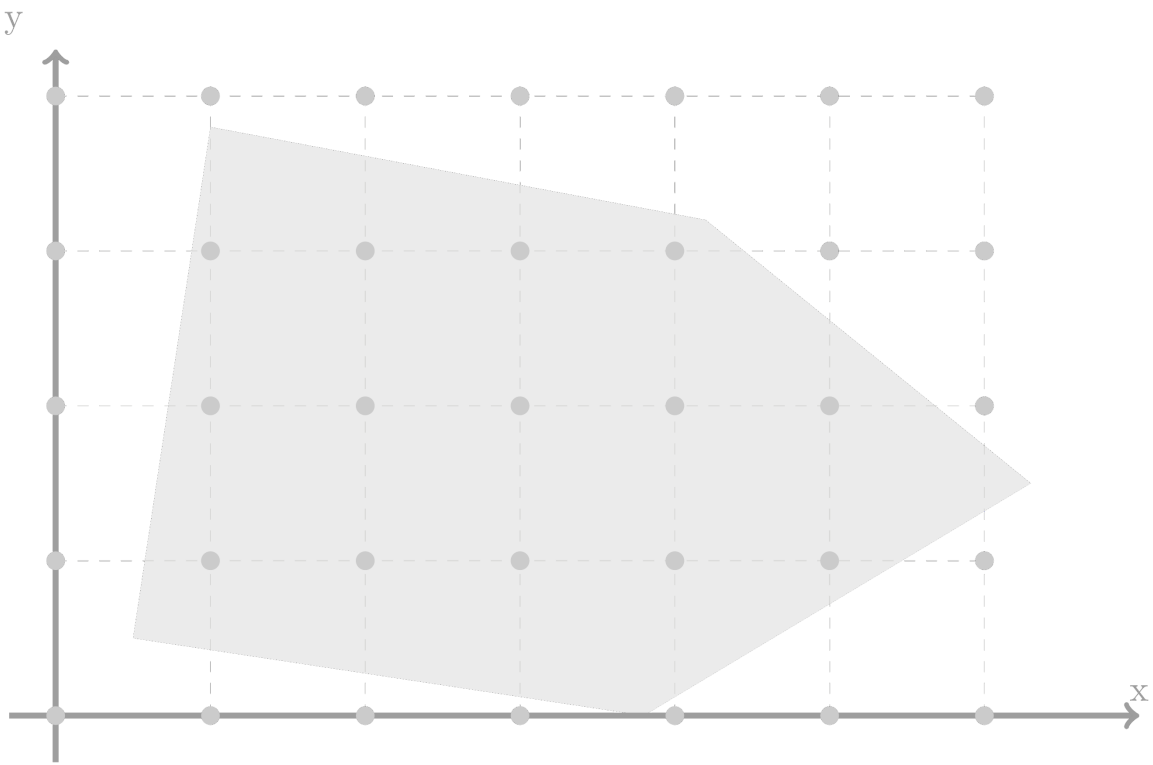
```
29  H20108 11407                          1.475012e+09 8.0000e+08  45.8%    4.2
30   20707 11751 1.0000e+09   44    40 1.4750e+09 8.0000e+08  45.8%    4.5
31   38874 21313 1.0500e+09   56    33 1.4750e+09 8.0000e+08  45.8%    4.8
32  *43317 22917              61     1.466682e+09 8.0000e+08  45.5%    5.5
33   44236 23753 1.1000e+09   39    44 1.4667e+09 8.0000e+08  45.5%    5.7
34  H44555 22869                          1.400014e+09 8.0000e+08  42.9%    5.7
35  H46604 23057                          1.375015e+09 8.0000e+08  41.8%    6.0
36  *53603 23808              37     1.366680e+09 8.0000e+08  41.5%    5.7
37  H54046 20392                          1.200013e+09 8.0000e+08  33.3%    5.8
38
39  Explored 62989 nodes (369883 simplex iterations) in 18.50 seconds (12.0
40  Thread count was 8 (of 8 available processors)
41
42  Solution count 10: 1.20001e+09 1.36668e+09 1.37501e+09 ... 1.51001e+09
43
44  Optimal solution found (tolerance 1.00e-04)
45  Best objective 1.200012600000e+09, best bound 1.200012600000e+09, gap 0
```

Tune Solver Parameters

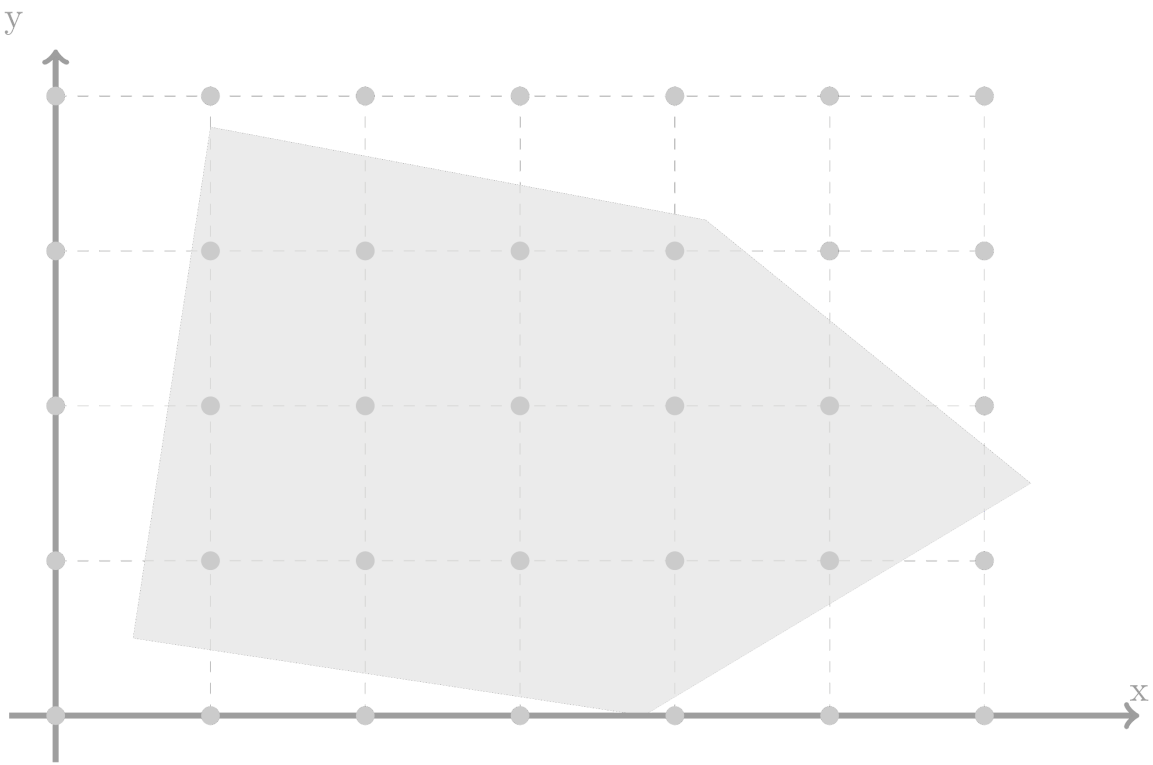
Example: glass4

```
29  H20108 11407                          1.475012e+09 8.0000e+08  45.8%   4.2
30   20707 11751 1.0000e+09   44   40 1.4750e+09 8.0000e+08  45.8%   4.5
31   38874 21313 1.0500e+09   56   33 1.4750e+09 8.0000e+08  45.8%   4.8
32  *43317 22917              61    1.466682e+09 8.0000e+08  45.5%   5.5
33   44236 23753 1.1000e+09   39   44 1.4667e+09 8.0000e+08  45.5%   5.7
34  H44555 22869                          1.400014e+09 8.0000e+08  42.9%   5.7
35  H46604 23057                          1.375015e+09 8.0000e+08  41.8%   6.0
36  *53603 23808              37    1.366680e+09 8.0000e+08  41.5%   5.7
37  H54046 20392                          1.200013e+09 8.0000e+08  33.3%   5.8
38
39  Explored 62989 nodes (369883 simplex iterations) in 18.50 seconds (12.0
40  Thread count was 8 (of 8 available processors)
41
42  Solution count 10: 1.20001e+09 1.36668e+09 1.37501e+09 ... 1.51001e+09
43
44  Optimal solution found (tolerance 1.00e-04)
45  Best objective 1.200012600000e+09, best bound 1.200012600000e+09, gap 0
```

Geometry of MIP

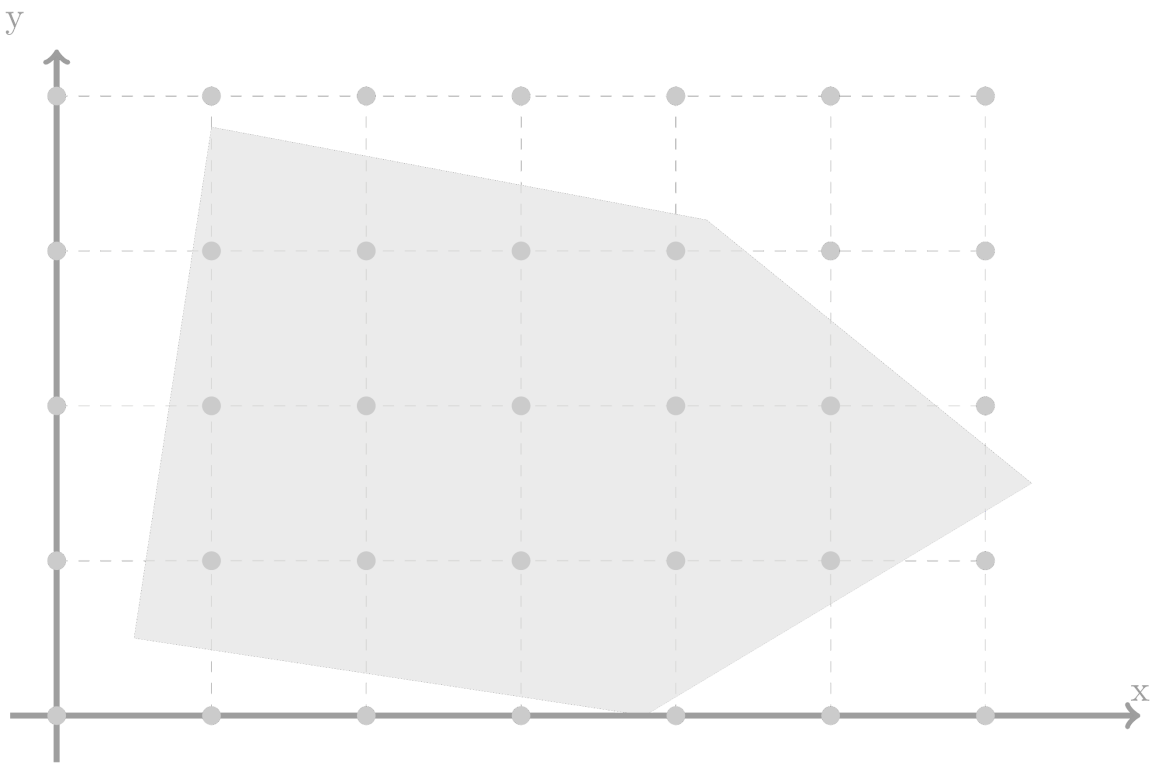


Geometry of MIP



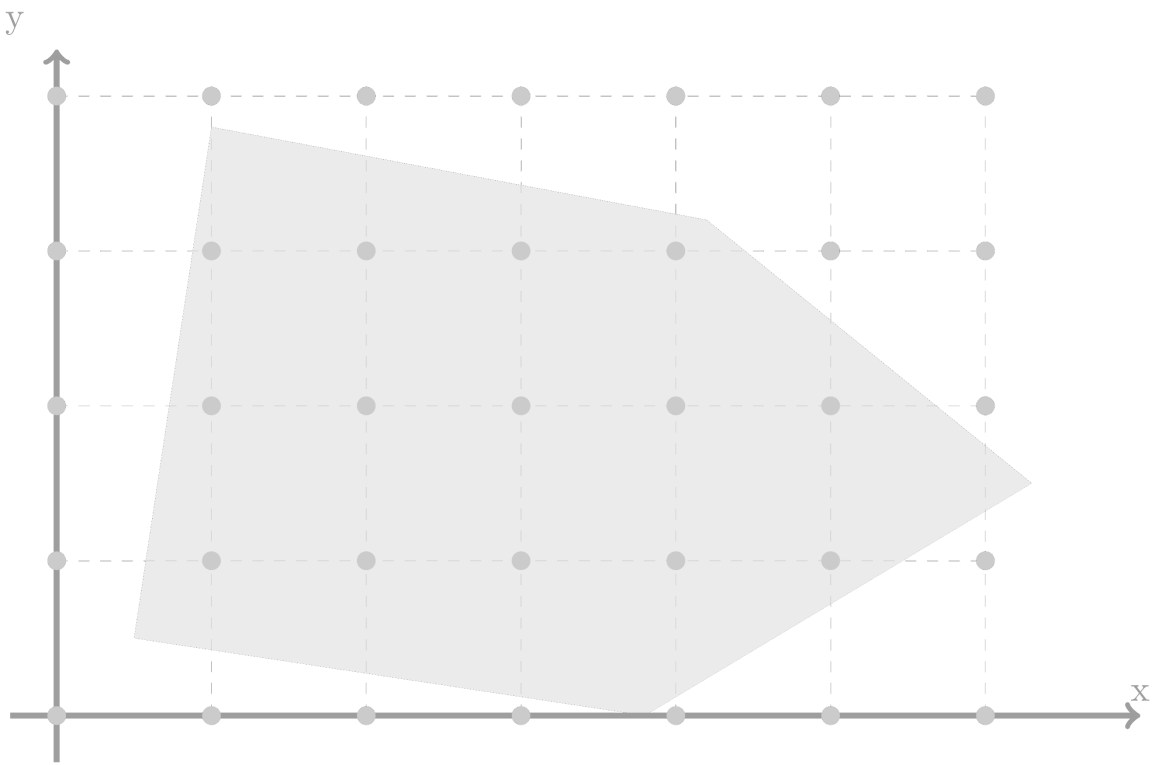
- 2 integer variables x and y

Geometry of MIP



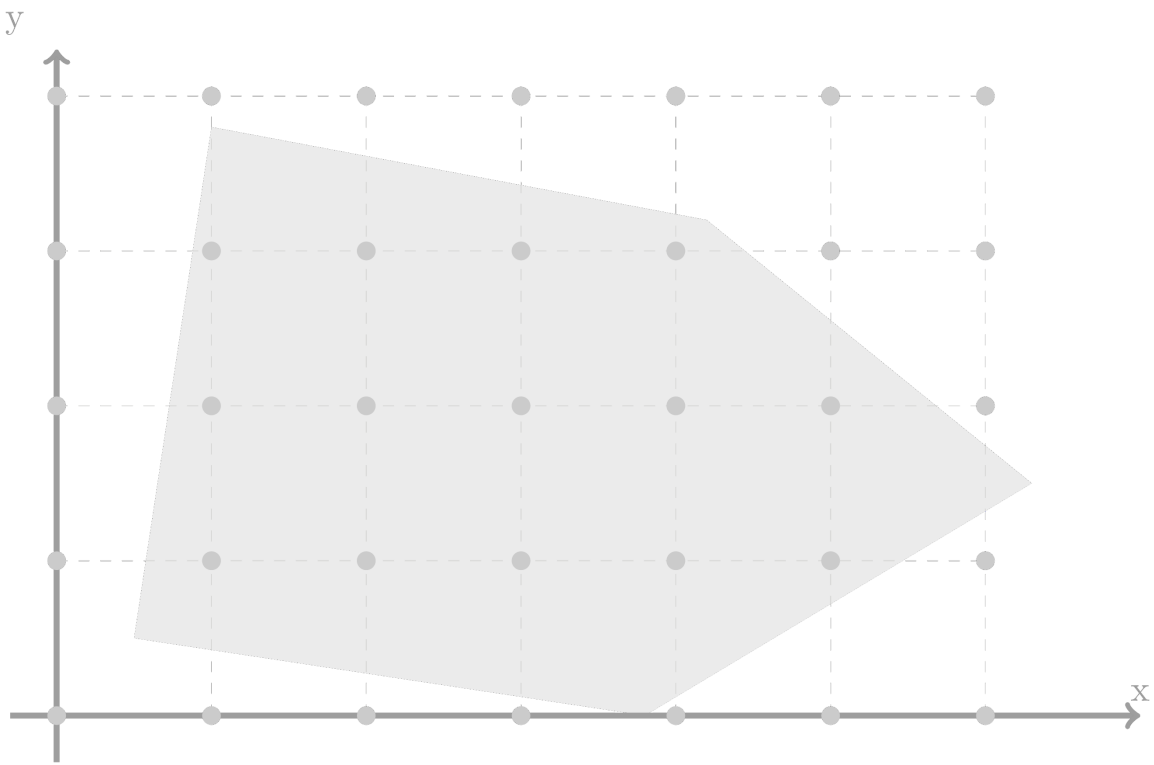
- 2 integer variables x and y
- 5 linear constraints define gray area = continuous or LP relaxation

Geometry of MIP



- 2 integer variables x and y
- 5 linear constraints define gray area = continuous or LP relaxation
- Feasible solutions are integer points inside the gray area

Geometry of MIP

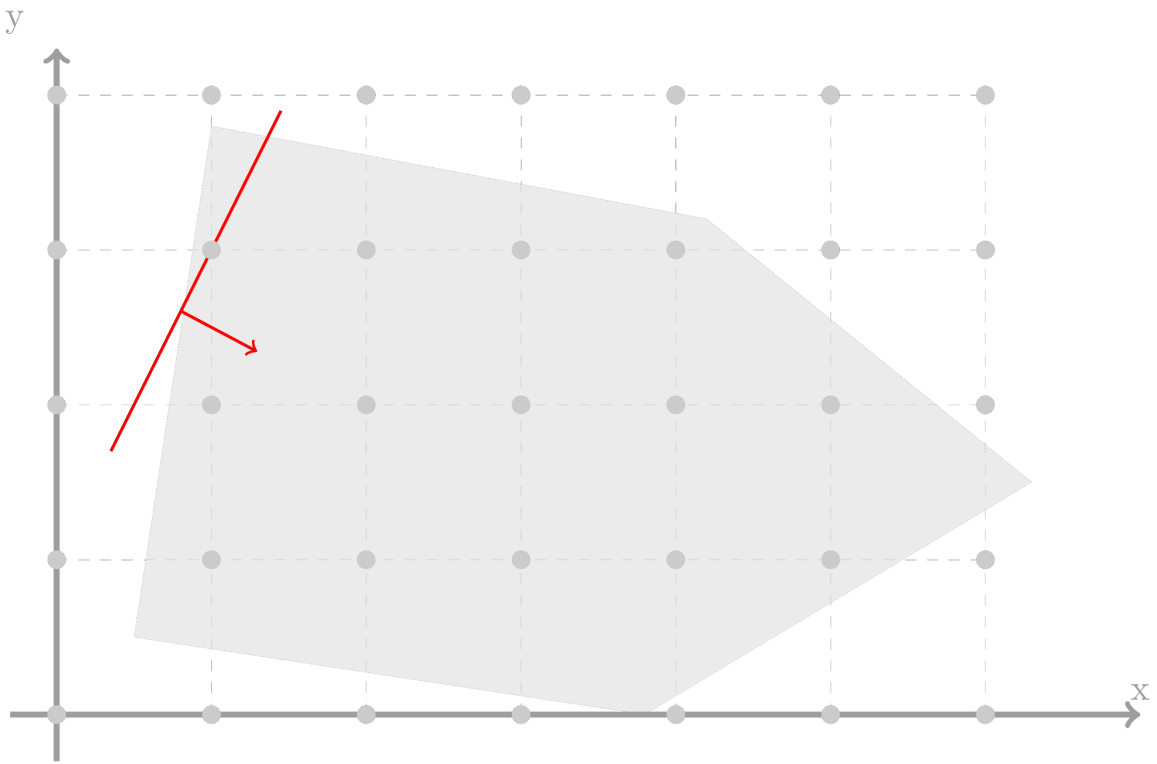


- 2 integer variables x and y
- 5 linear constraints define gray area = continuous or LP relaxation
- Feasible solutions are integer points inside the gray area
- Solvers heavily rely on LP relaxation, e.g., in bounds, heuristics, branching

Geometry of MIP

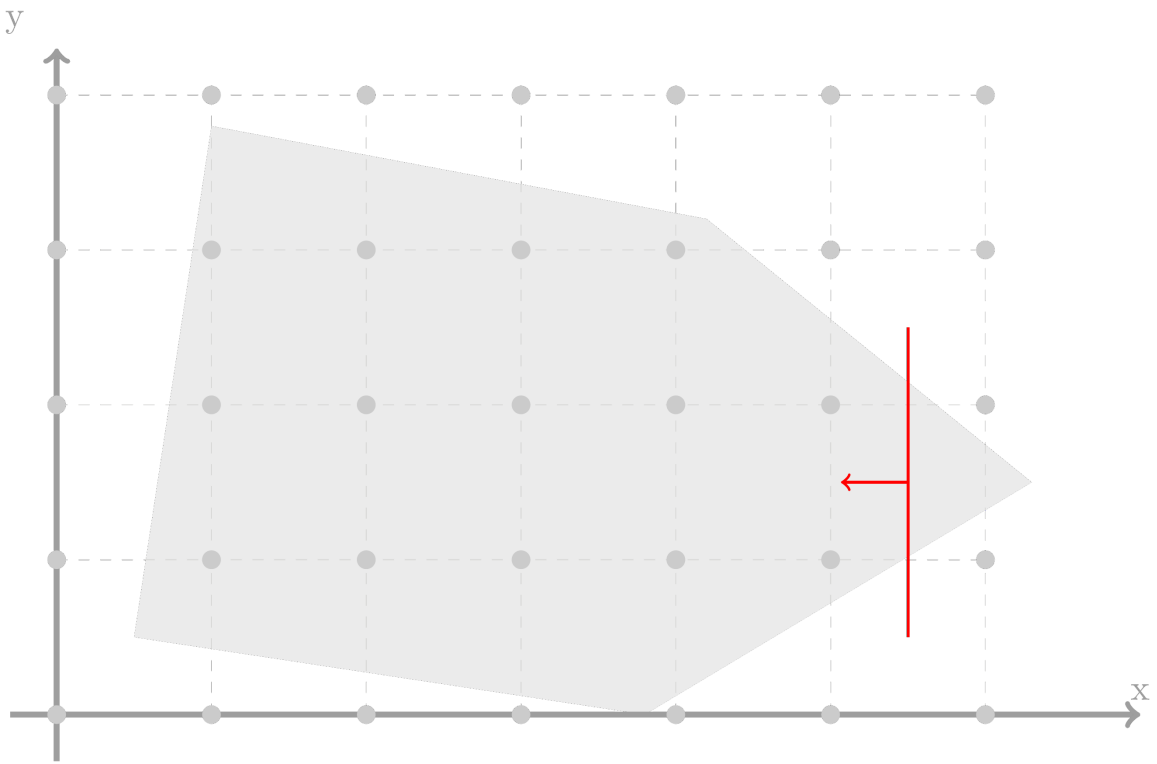
- Tighten the LP relaxation by adding constraints = Cut
- Set of feasible solutions stays the same
- Gray area gets smaller
- Variable bounds are also cuts

Geometry of MIP



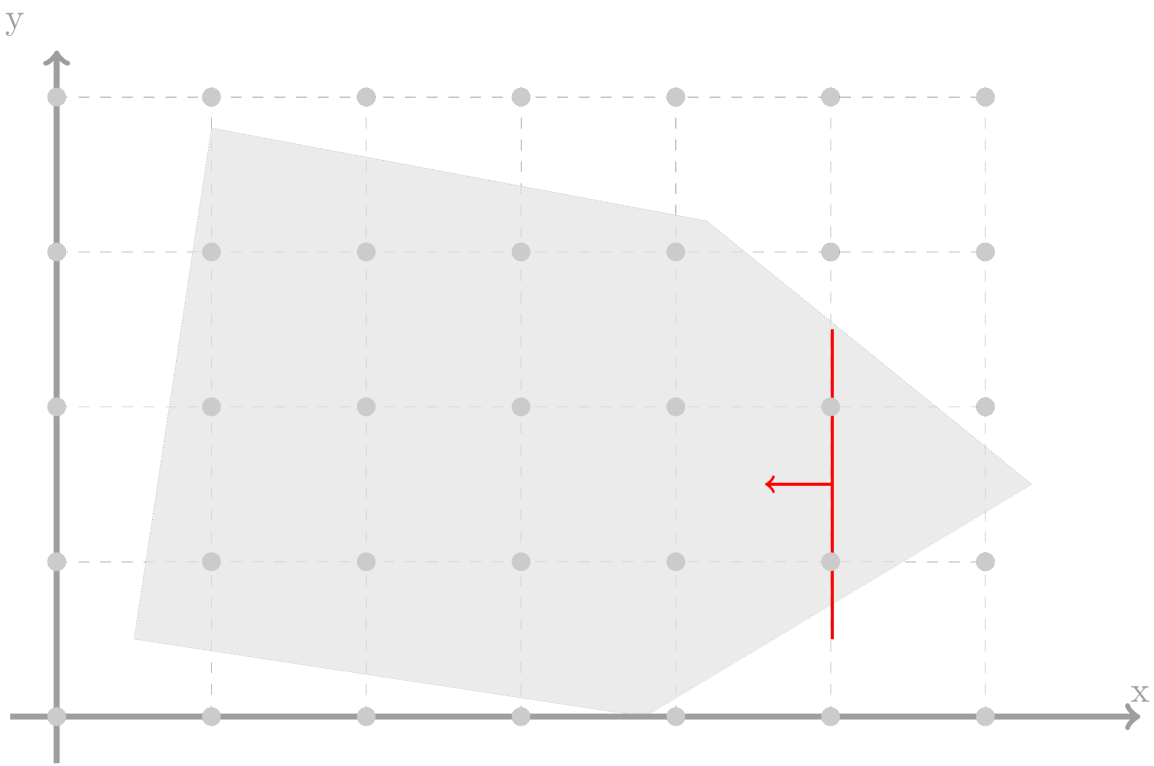
- Tighten the LP relaxation by adding constraints = Cut
- Set of feasible solutions stays the same
- Gray area gets smaller
- Variable bounds are also cuts

Geometry of MIP



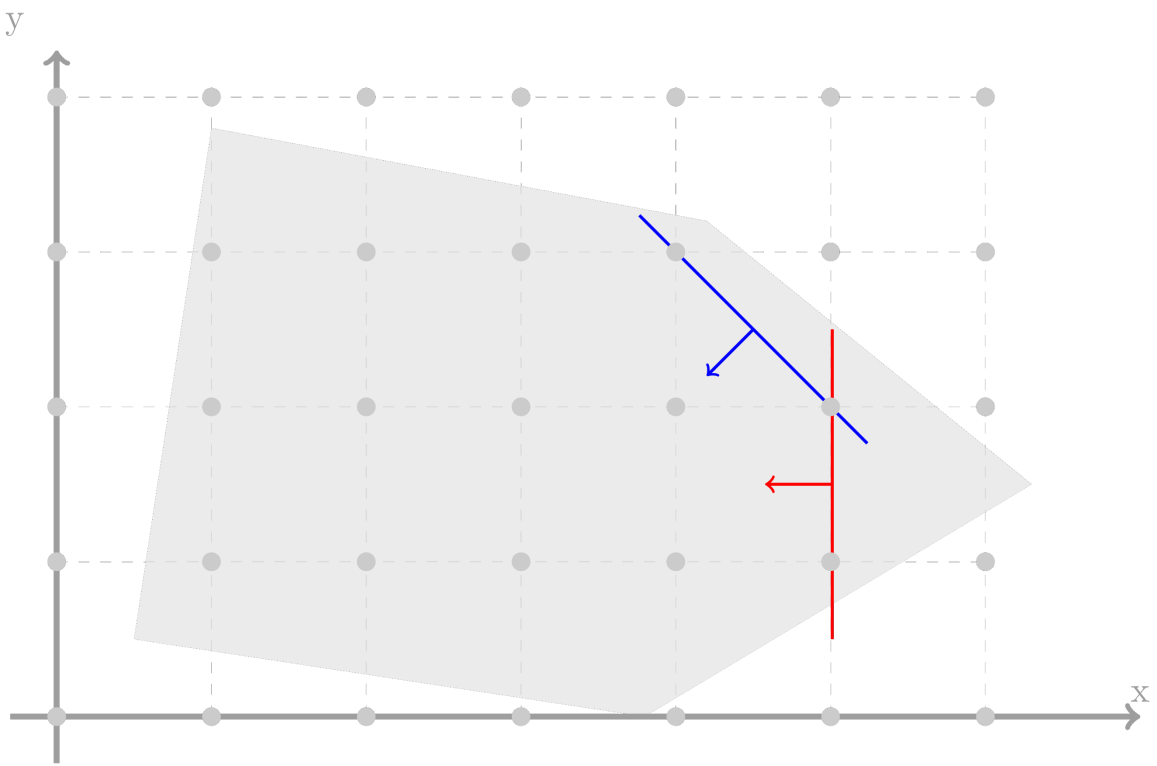
- Tighten the LP relaxation by adding constraints = Cut
- Set of feasible solutions stays the same
- Gray area gets smaller
- Variable bounds are also cuts

Geometry of MIP



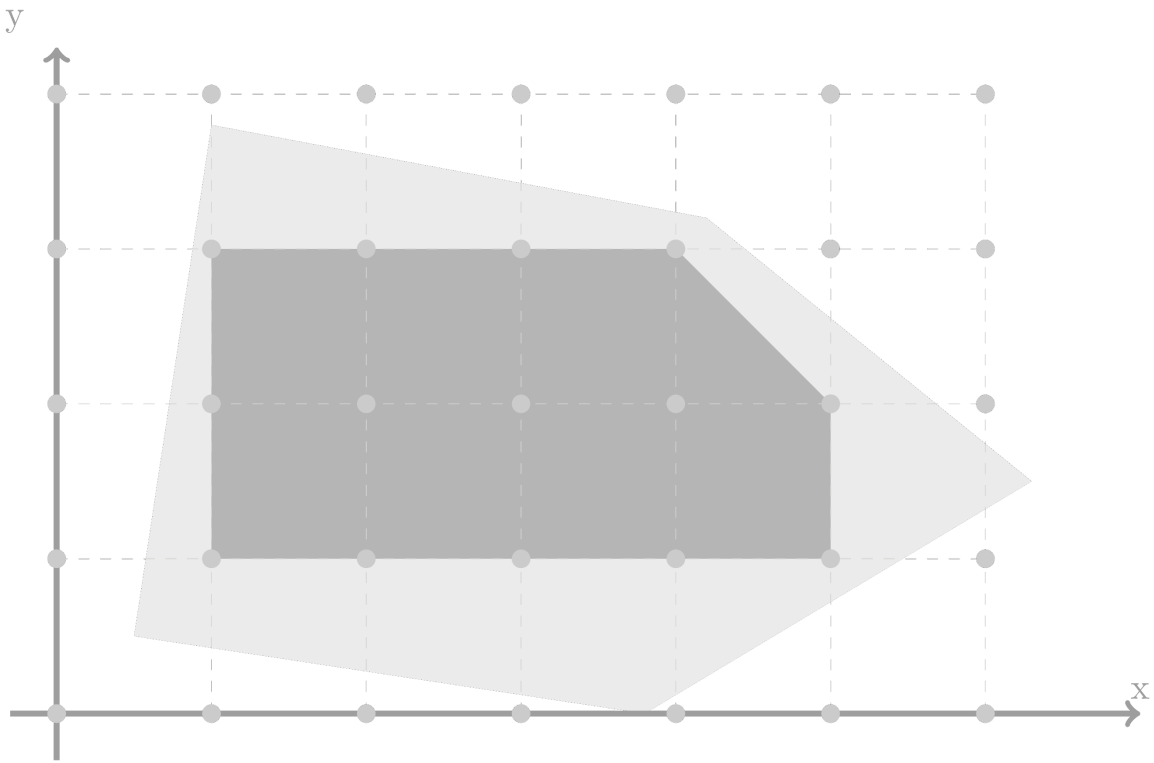
- Tighten the LP relaxation by adding constraints = Cut
- Set of feasible solutions stays the same
- Gray area gets smaller
- Variable bounds are also cuts

Geometry of MIP

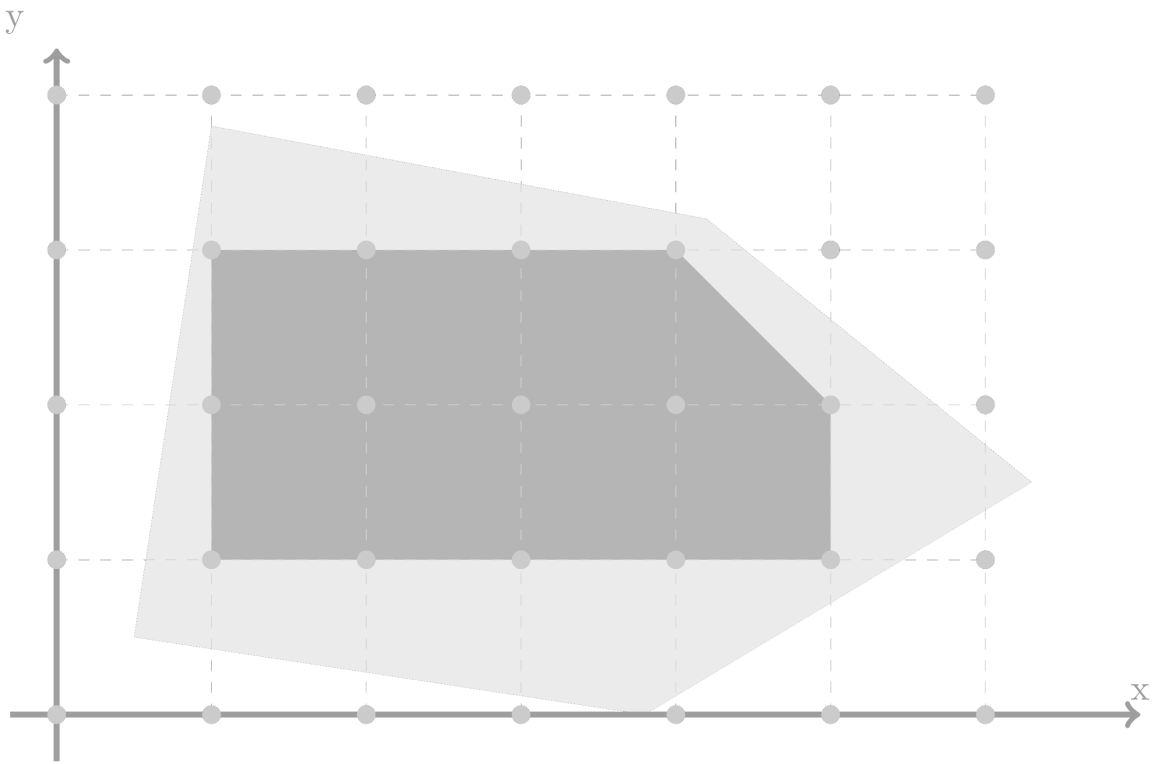


- Tighten the LP relaxation by adding constraints = Cut
- Set of feasible solutions stays the same
- Gray area gets smaller
- Variable bounds are also cuts

Geometry of MIP

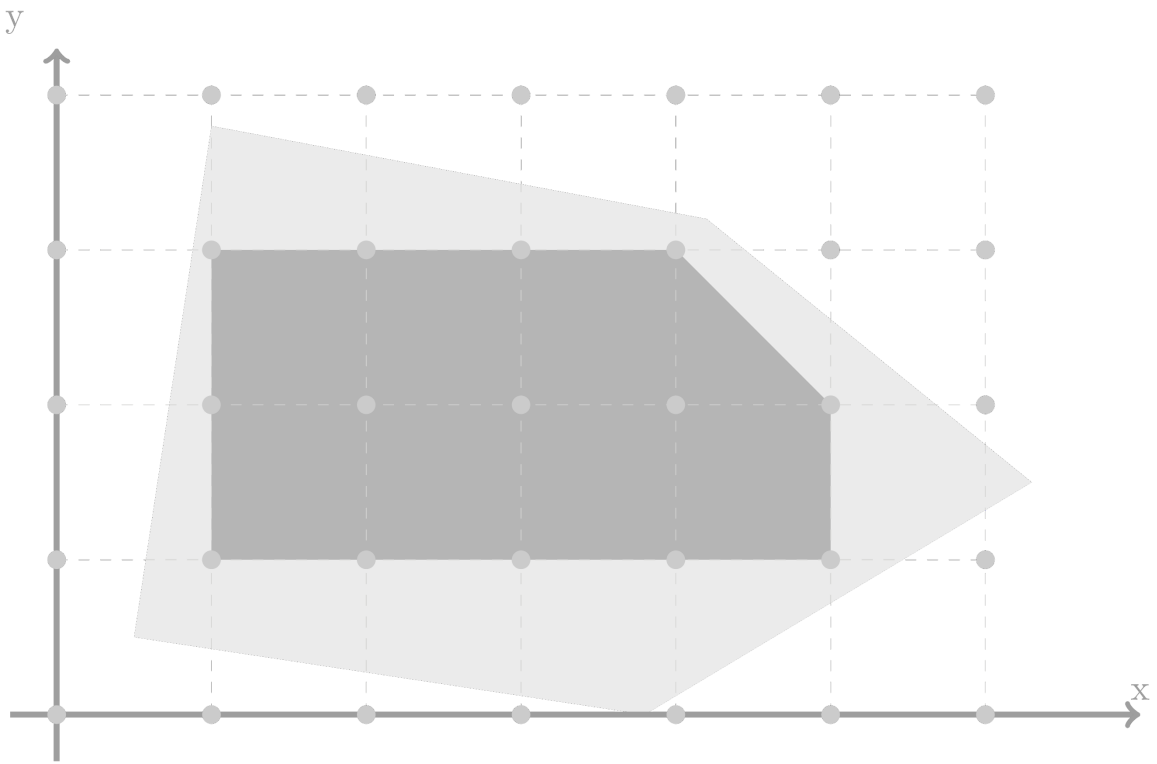


Geometry of MIP



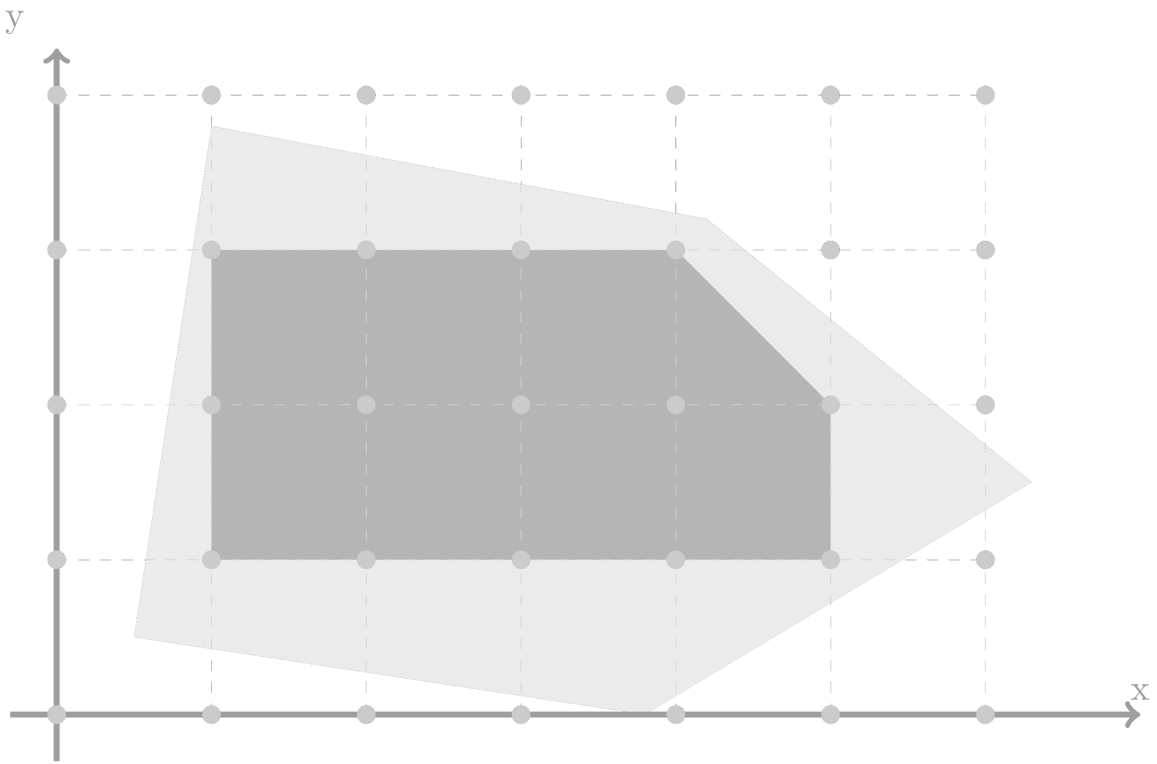
- Dark gray area = ideal formulation / convex hull

Geometry of MIP



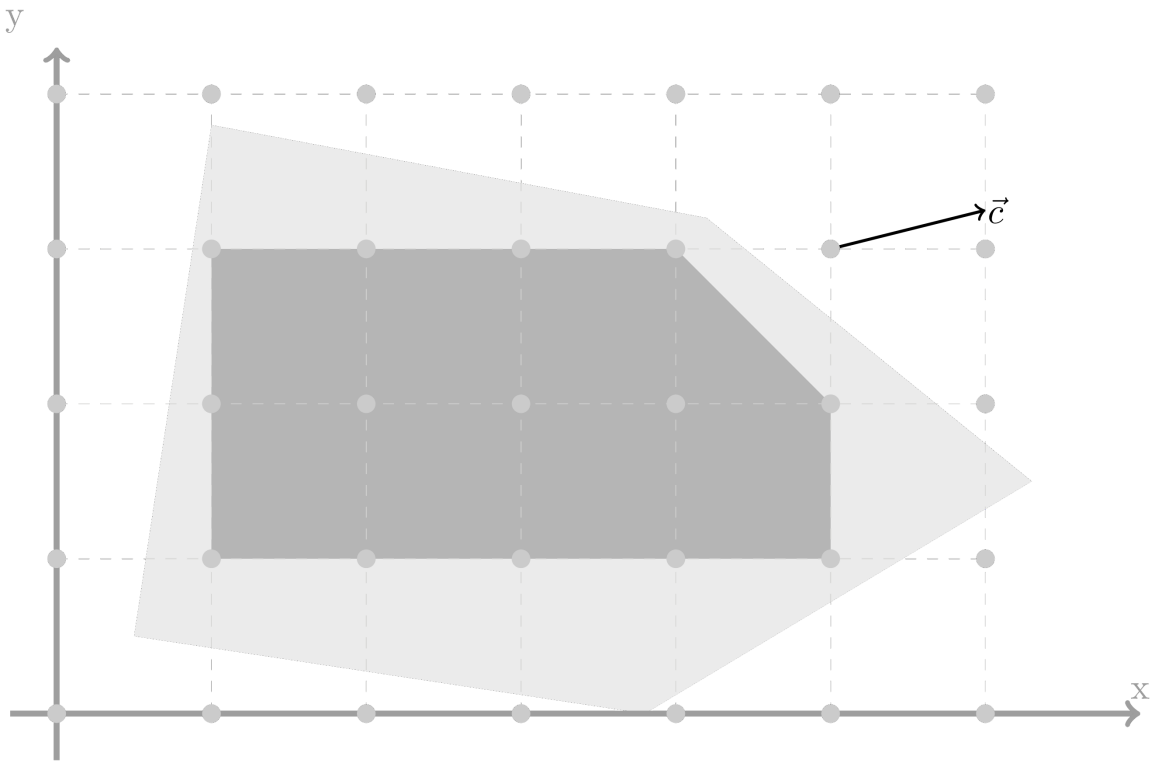
- Dark gray area = ideal formulation / convex hull
- Solver cuts and presolve try to find convex hull

Geometry of MIP



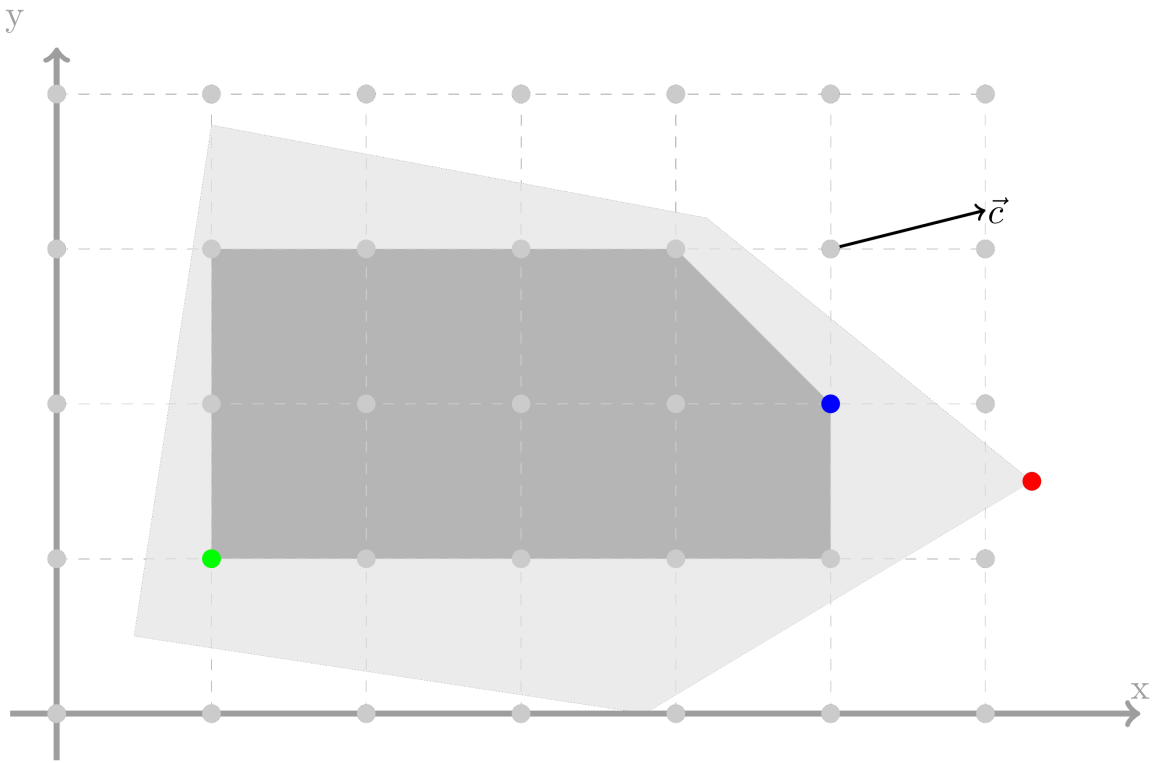
- Dark gray area = ideal formulation / convex hull
- Solver cuts and presolve try to find convex hull
- Why ideal? MIP reduces to LP

Geometry of MIP



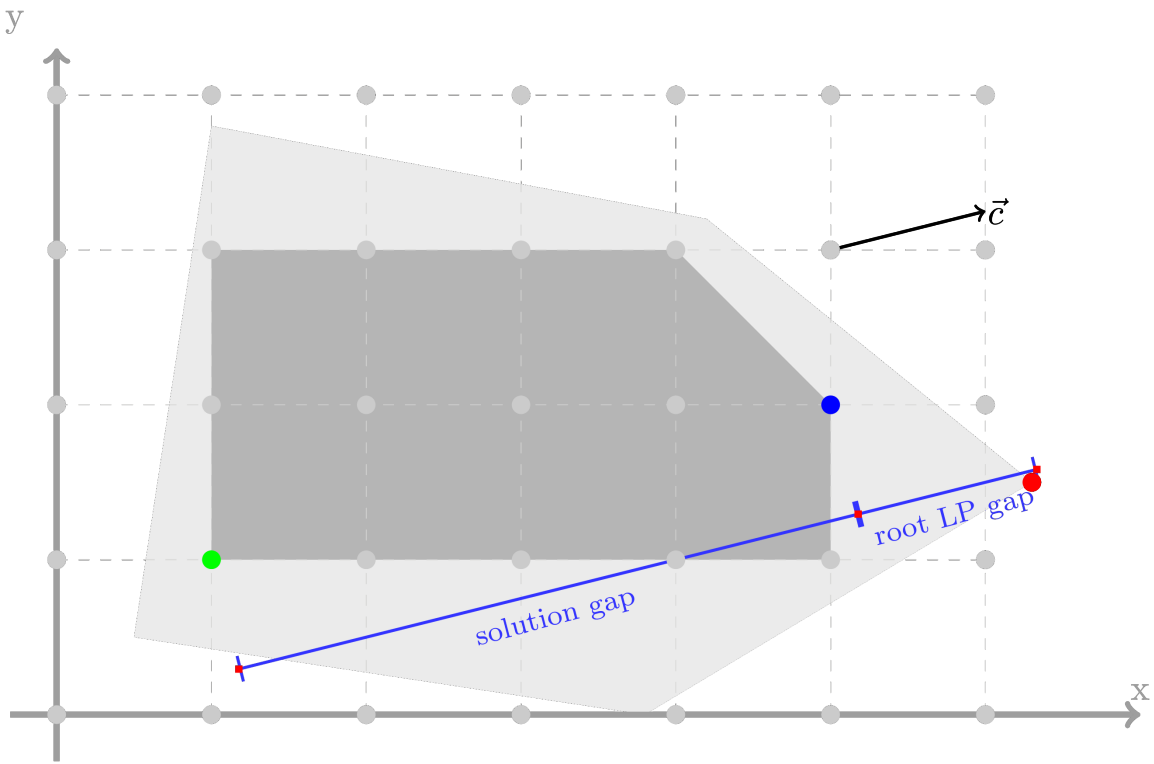
- Objective function vector c

Geometry of MIP



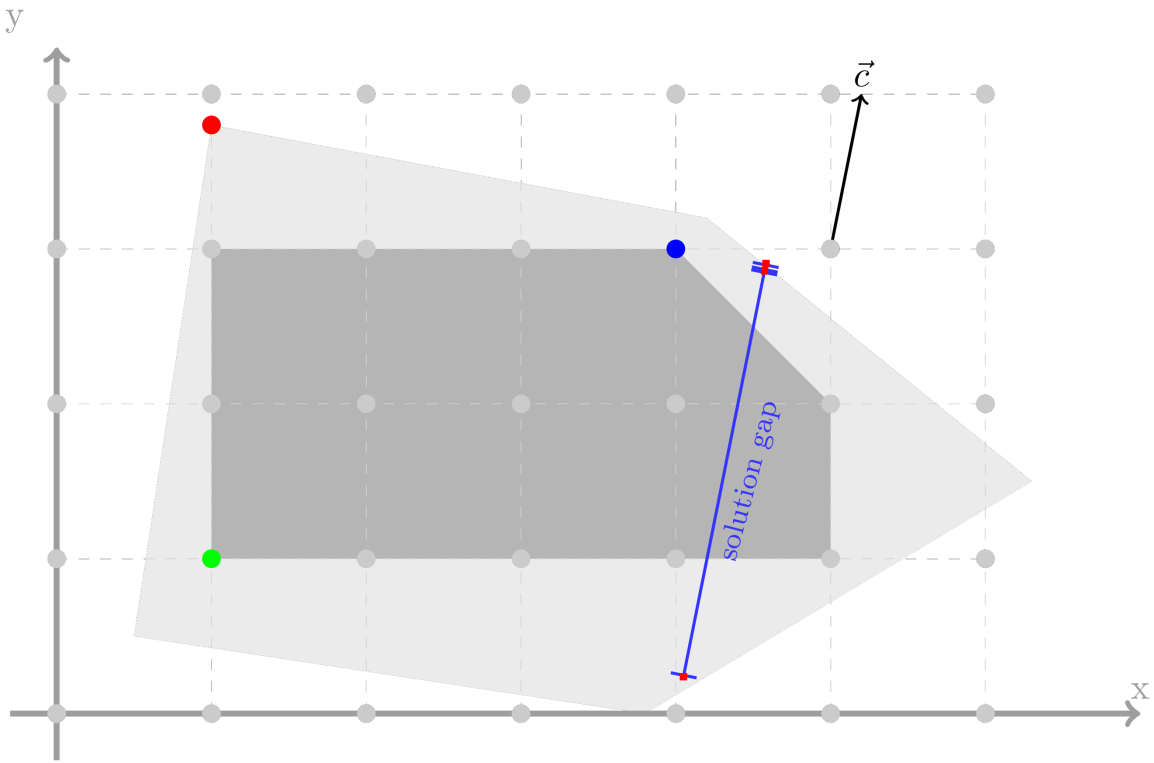
- Potential initial solution
- Optimal integer solution
- Optimal LP relaxation solution

Geometry of MIP



- Potential initial solution
- Optimal integer solution
- Optimal LP relaxation solution

Geometry of MIP



- Potential initial solution
- Optimal integer solution
- Optimal LP relaxation solution
- Gaps are highly dependent on objective function
- Even small root LP gaps can hide bad formulations

Geometry of MIP
Why is this important?

Geometry of MIP
Why is this important?

- Integer problems can have many formulations for the same set of solutions, which one should you choose?

Geometry of MIP
Why is this important?

- Integer problems can have many formulations for the same set of solutions, which one should you choose?
- Tighter formulations often lead to better performance, even if the model has more variables and/or constraints

Geometry of MIP
Why is this important?

- Integer problems can have many formulations for the same set of solutions, which one should you choose?
- Tighter formulations often lead to better performance, even if the model has more variables and/or constraints
- Art of Modeling

Geometry of MIP
Why is this important?

- Integer problems can have many formulations for the same set of solutions, which one should you choose?
- Tighter formulations often lead to better performance, even if the model has more variables and/or constraints
- Art of Modeling
  - Identify and understand strength

Geometry of MIP
Why is this important?

- Integer problems can have many formulations for the same set of solutions, which one should you choose?
- Tighter formulations often lead to better performance, even if the model has more variables and/or constraints
- Art of Modeling
  - Identify and understand strength
  - Find cuts

Geometry of MIP
Why is this important?

- Integer problems can have many formulations for the same set of solutions, which one should you choose?
- Tighter formulations often lead to better performance, even if the model has more variables and/or constraints
- Art of Modeling
  - Identify and understand strength
  - Find cuts
  - Reformulate

Example: Min-max

$$\begin{aligned}
\min \quad & t \\
& x_j \le t && \forall j = 1, \dots, n \\
& \sum_{j=1}^{n} x_j = 1 \\
& x_j \in \{0, 1\} && \forall j = 1, \dots, n
\end{aligned}$$

Example: Min-max

$$
\begin{aligned}
\min \quad & t & & \\
& x_j \le t & & \forall j = 1, \ldots, n \\
& \sum_{j=1}^{n} x_j = 1 & & \\
& x_j \in \{0, 1\} & & \forall j = 1, \ldots, n
\end{aligned}
$$

- Optimal MIP solution: $t = 1,\ x_j = 1$, for one particular j

Example: Min-max

$$
\begin{aligned}
\min \quad & t & & \\
& x_j \le t & & \forall j = 1, \ldots, n \\
& \sum_{j=1}^{n} x_j = 1 & & \\
& x_j \in \{0, 1\} & & \forall j = 1, \ldots, n
\end{aligned}
$$

- Optimal MIP solution: $t = 1,\ x_j = 1$, for one particular j
- LP relaxation solution: $t = \frac{1}{n},\ x_j = \frac{1}{n}$, for **ALL** $j \rightarrow$ bound is weak

Example: Min-max

$$\begin{aligned} \min \quad & t \\ & x_j \le t && \forall j = 1, \dots, n \\ & \sum_{j=1}^{n} x_j = 1 \\ & x_j \in \{0, 1\} && \forall j = 1, \dots, n \end{aligned}$$

- Optimal MIP solution: $t = 1,\ x_j = 1$, for one particular j
- LP relaxation solution: $t = \frac{1}{n},\ x_j = \frac{1}{n}$, for **ALL** $j \rightarrow$ bound is weak
- LP and MIP solutions have an inherent disconnect, even worse for growing n

Example: Min-max

$$\begin{aligned} \min \quad & t \\ & x_j \le t && \forall j = 1, \dots, n \\ & \sum_{j=1}^{n} x_j = 1 \\ & x_j \in \{0, 1\} && \forall j = 1, \dots, n \end{aligned}$$

- Optimal MIP solution: $t = 1,\ x_j = 1$, for one particular j
- LP relaxation solution: $t = \frac{1}{n},\ x_j = \frac{1}{n}$, for **ALL** $j \rightarrow$ bound is weak
- LP and MIP solutions have an inherent disconnect, even worse for growing n
- In general, try to push up t as much as possible

Example: Min-max

$$\begin{aligned} \min \quad & t \\ & x_j \le t && \forall j = 1, \dots, n \\ & \sum_{j=1}^{n} x_j = 1 \\ & x_j \in \{0, 1\} && \forall j = 1, \dots, n \end{aligned}$$

- Optimal MIP solution: $t = 1$, $x_j = 1$, for one particular j
- LP relaxation solution: $t = \frac{1}{n}$, $x_j = \frac{1}{n}$, for **ALL** $j \rightarrow$ bound is weak
- LP and MIP solutions have an inherent disconnect, even worse for growing n
- In general, try to push up t as much as possible
- Here, we can add constraint $\sum_{j=1}^{n} x_j \le t$

Example: Facility Location

Example: Facility Location

- **Input data**:

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets
  - W: Set of candidate warehouse locations

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets
  - W: Set of candidate warehouse locations
  - $\mathrm{f_w}$: Fixed cost for building warehouse w

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets
  - W: Set of candidate warehouse locations
  - f_w: Fixed cost for building warehouse w
  - c_{ws}: Cost of supplying supermarket s from warehouse w

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets
  - W: Set of candidate warehouse locations
  - f_w: Fixed cost for building warehouse w
  - c_{ws}: Cost of supplying supermarket s from warehouse w
- **Choose a subset of warehouses to supply all supermarkets, with minimal total building and supply costs**

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets
  - W: Set of candidate warehouse locations
  - f_w: Fixed cost for building warehouse w
  - c_{ws}: Cost of supplying supermarket s from warehouse w
- **Choose a subset of warehouses to supply all supermarkets, with minimal total building and supply costs**
- **Decision Variables**:

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets
  - W: Set of candidate warehouse locations
  - f_w: Fixed cost for building warehouse w
  - c_{ws}: Cost of supplying supermarket s from warehouse w
- **Choose a subset of warehouses to supply all supermarkets, with minimal total building and supply costs**
- **Decision Variables**:
  - $y_w \in \{0, 1\}$: Build warehouse w ($y_w = 1$) or not ($y_w = 0$)

Example: Facility Location

- **Input data**:
  - S: Set of supermarkets
  - W: Set of candidate warehouse locations
  - f_w: Fixed cost for building warehouse w
  - c_{ws}: Cost of supplying supermarket s from warehouse w
- **Choose a subset of warehouses to supply all supermarkets, with minimal total building and supply costs**
- **Decision Variables**:
  - $y_w \in \{0, 1\}$: Build warehouse w ($y_w = 1$) or not ($y_w = 0$)
  - $0 \leq x_{ws} \leq 1$: Fraction of demand of supermarket s served by warehouse w

Example: Facility Location
Model

Example: Facility Location
Model

- **Objective**: Minimize total warehouse building and supply costs

$$\text{Min} \quad \sum_{w \in W} f_w \cdot y_w + \sum_{w \in W} \sum_{s \in S} c_{ws} \cdot x_{ws}$$

Example: Facility Location
Model

- **Objective**: Minimize total warehouse building and supply costs

$$\text{Min} \quad \sum_{w \in W} f_w \cdot y_w + \sum_{w \in W} \sum_{s \in S} c_{ws} \cdot x_{ws}$$

- Each supermarket has to be fully supplied, i.e., the sum of fractions received from each warehouse must be equal to 1.

$$\sum_{w \in W} x_{ws} = 1 \quad \forall s \in S$$

Example: Facility Location
Model

- **Objective**: Minimize total warehouse building and supply costs

$$\text{Min} \quad \sum_{w \in W} f_w \cdot y_w + \sum_{w \in W} \sum_{s \in S} c_{ws} \cdot x_{ws}$$

- Each supermarket has to be fully supplied, i.e., the sum of fractions received from each warehouse must be equal to 1.

$$\sum_{w \in W} x_{ws} = 1 \quad \forall s \in S$$

- Warehouse w has to be opened if at least one supermarket is supplied.

$$\sum_{s \in S} x_{ws} \leq |S| \cdot y_w \quad \forall w \in W$$

Example: Facility Location

200 warehouse locations, 2000 supermarkets

```
Optimize a model with 2200 rows, 400200 columns and 800200 nonzeros
Model fingerprint: 0x889c7bde
Variable types: 400000 continuous, 200 integer (200 binary)
Coefficient statistics:
  Matrix range     [1e+00, 2e+03]
  Objective range  [1e+00, 4e+03]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 1657011.0000
Presolve time: 0.89s
Presolved: 2200 rows, 400200 columns, 800200 nonzeros
Variable types: 400000 continuous, 200 integer (200 binary)

Root relaxation: objective 7.496728e+04, 0 iterations, 0.57 seconds (0.

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node
```

Example: Facility Location

200 warehouse locations, 2000 supermarkets

```
  Objective range  [1e+00, 4e+03]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 1657011.0000
Presolve time: 0.89s
Presolved: 2200 rows, 400200 columns, 800200 nonzeros
Variable types: 400000 continuous, 200 integer (200 binary)

Root relaxation: objective 7.496728e+04, 0 iterations, 0.57 seconds (0.

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

H    0     0                    655075.00000 74967.2750  88.6%     -
H    0     0                    653009.00000 74967.2750  88.5%     -
H    0     0                    649303.00000 74967.2750  88.5%     -
H    0     0                    315273.00000 74967.2750  76.2%     -
     0     0 90077.2092    0  189 315273.000 90077.2092  71.4%
```

Example: Facility Location

200 warehouse locations, 2000 supermarkets

```
Presolved: 2200 rows, 400200 columns, 800200 nonzeros
Variable types: 400000 continuous, 200 integer (200 binary)

Root relaxation: objective 7.496728e+04, 0 iterations, 0.57 seconds (0.

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

H    0     0                       655075.00000 74967.2750  88.6%     -
H    0     0                       653009.00000 74967.2750  88.5%     -
H    0     0                       649303.00000 74967.2750  88.5%     -
H    0     0                       315273.00000 74967.2750  76.2%     -
     0     0 90077.2092    0  189 315273.000 90077.2092  71.4%     -
H    0     0                       309486.00000 113880.544  63.2%     -
     0     0 113880.637    0  199 309486.000 113880.637  63.2%     -
     0     0 113880.637    0  199 309486.000 113880.637  63.2%     -
H    0     0                       288245.00000 113880.637  60.5%     -
     0     0 127672.131    0  198 288245.000 127672.131  55.7%
```

Example: Facility Location

200 warehouse locations, 2000 supermarkets

```
 95     826    791 203413.089    47   150 218823.000 197475.343  9.76%  33.5   3
 96     854    816 203917.562    51   147 218823.000 197475.343  9.76%  33.6   3
 97     879    839 204990.022    60   138 218823.000 197475.343  9.76%  34.0   3
 98     902    876 207260.562    66   132 218823.000 197475.343  9.76%  34.6   3
 99     939    911 208445.648    79   119 218823.000 197475.343  9.76%  34.7   3
100     974    952 209719.263    88   110 218823.000 197475.343  9.76%  35.0   3
101    1015    999 211292.539    95   102 218823.000 197475.343  9.76%  35.4   3
102    1063   1000 213064.979    75   196 218823.000 197475.343  9.76%  36.1   3
103  * 1064    950              12    215114.00000 215114.000  0.00%  36.0   4
104
105  Explored 1064 nodes (62733 simplex iterations) in 407.36 seconds (209.7
106  Thread count was 8 (of 8 available processors)
107
108  Solution count 10: 215114 218823 218836 ... 238519
109
110  Optimal solution found (tolerance 1.00e-04)
111  Best objective 2.151140000000e+05, best bound 2.151140000000e+05, gap 0
```

Example: Facility Location
Cuts

Example: Facility Location
Cuts

- Warehouse w has to be opened if at least one supermarket is supplied.

$$\sum_{s \in S} x_{ws} \leq |S| \cdot y_w \qquad \forall w \in W$$

Example: Facility Location
Cuts

- Warehouse w has to be opened if at least one supermarket is supplied.

$$\sum_{s \in S} x_{ws} \leq |S| \cdot y_w \qquad \forall w \in W$$

- In most solutions, the left-hand side will be much smaller than $|S| \cdot y_w$

Example: Facility Location
Cuts

- Warehouse w has to be opened if at least one supermarket is supplied.

$$\sum_{s \in S} x_{ws} \leq |S| \cdot y_w \quad \forall w \in W$$

- In most solutions, the left-hand side will be much smaller than $|S| \cdot y_w$
- In the LP relaxation, due to the large Big-M value $|S|$ and since we minimize $f_w \cdot y_w$, the value for y_w can be set to a small fractional value

Example: Facility Location
Cuts

- Warehouse w has to be opened if at least one supermarket is supplied.

$$\sum_{s \in S} x_{ws} \leq |S| \cdot y_w \quad \forall w \in W$$

- In most solutions, the left-hand side will be much smaller than $|S| \cdot y_w$
- In the LP relaxation, due to the large Big-M value $|S|$ and since we minimize $f_w \cdot y_w$, the value for y_w can be set to a small fractional value
- **Constraint Disaggregation**: If any single supermarket is supplied by a warehouse, the warehouse has to be opened.

$$x_{ws} \leq y_w \quad \forall w \in W, \forall s \in S$$

Example: Facility Location

```
Optimize a model with 402000 rows, 400200 columns and 1200000 nonzeros
Model fingerprint: 0xab88c055
Variable types: 400000 continuous, 200 integer (200 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+00, 4e+03]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 1657011.0000
Presolve time: 2.35s
Presolved: 402000 rows, 400200 columns, 1200000 nonzeros
Variable types: 400000 continuous, 200 integer (200 binary)
Deterministic concurrent LP optimizer: primal simplex, dual simplex, an
Showing barrier log only...

Root barrier log...

Ordering time: 0.18s
```

Example: Facility Location

```
44
45 Barrier performed 14 iterations in 13.93 seconds (4.14 work units)
46 Barrier solve interrupted - model solved by another algorithm
47
48 Concurrent spin time: 1.23s (can be avoided by choosing Method=3)
49
50 Solved with dual simplex
51
52 Root relaxation: objective 2.151140e+05, 13089 iterations, 11.13 second
53
54     Nodes    |    Current Node    |     Objective Bounds      |     Wor
55  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node
56
57 *    0     0               0    215114.00000 215114.000  0.00%     -
58
59 Explored 1 nodes (13089 simplex iterations) in 14.05 seconds (4.26 work
60 Thread count was 8 (of 8 available processors)
61
```

Example: Facility Location

```
Concurrent spin time: 1.25s (can be avoided by choosing Method=3)

Solved with dual simplex

Root relaxation: objective 2.151140e+05, 13089 iterations, 11.13 second

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

*    0     0               0    215114.00000 215114.000  0.00%     -

Explored 1 nodes (13089 simplex iterations) in 14.05 seconds (4.26 work
Thread count was 8 (of 8 available processors)

Solution count 2: 215114 1.65701e+06

Optimal solution found (tolerance 1.00e-04)
Best objective 2.151140000000e+05, best bound 2.151140000000e+05, gap 0
```

Solver Heuristics

Solver Heuristics

- General-purpose = work for models from many different areas

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:
  1. Fix or round values for (some) integer variables based on

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:
  1. Fix or round values for (some) integer variables based on
     - Relaxation solutions

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:
  1. Fix or round values for (some) integer variables based on
     - Relaxation solutions
     - Available integer solutions

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:
  1. Fix or round values for (some) integer variables based on
     - Relaxation solutions
     - Available integer solutions
  2. Solve sub-MIP for the rest

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:
  1. Fix or round values for (some) integer variables based on
     - Relaxation solutions
     - Available integer solutions
  2. Solve sub-MIP for the rest
- Sometimes relaxations are hard to solve or do not give much information → NoRel heuristic

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:
  1. Fix or round values for (some) integer variables based on
     - Relaxation solutions
     - Available integer solutions
  2. Solve sub-MIP for the rest
- Sometimes relaxations are hard to solve or do not give much information → NoRel heuristic
  - Often works well for large binary models

Solver Heuristics

- General-purpose = work for models from many different areas
- Solver includes more than 30 different heuristics
- **Main concept**:
  1. Fix or round values for (some) integer variables based on
     - Relaxation solutions
     - Available integer solutions
  2. Solve sub-MIP for the rest
- Sometimes relaxations are hard to solve or do not give much information → NoRel heuristic
  - Often works well for large binary models
  - Activate via parameter `NoRelHeurTime` or `NoRelHeurWork`

NoRel Heuristic

Example: glass4

```
Set parameter Heuristics to value 0.5
Set parameter NoRelHeurTime to value 5
Set parameter Cuts to value 0
...
Optimize a model with 396 rows, 322 columns and 1815 nonzeros
Model fingerprint: 0x18b19fdf
Variable types: 20 continuous, 302 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 8e+06]
  Objective range  [1e+00, 1e+06]
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.00s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09
Starting NoRel heuristic
```

NoRel Heuristic

Example: glass4

```
  Objective range  [1e+00, 1e+06]
  Bounds range     [1e+00, 8e+02]
  RHS range        [1e+00, 8e+06]
Presolve removed 6 rows and 6 columns
Presolve time: 0.00s
Presolved: 390 rows, 316 columns, 1803 nonzeros
Variable types: 19 continuous, 297 integer (297 binary)
Found heuristic solution: objective 3.133356e+09
Starting NoRel heuristic
Found heuristic solution: objective 3.033354e+09
Found heuristic solution: objective 2.633356e+09
Found heuristic solution: objective 2.633356e+09
Found heuristic solution: objective 2.633355e+09
Found heuristic solution: objective 2.533354e+09
Found heuristic solution: objective 2.333352e+09
Found heuristic solution: objective 2.300021e+09
Found heuristic solution: objective 2.200019e+09
Found heuristic solution: objective 2.125017e+09
```

NoRel Heuristic

Example: glass4

```
Found heuristic solution: objective 1.650015e+09
Found heuristic solution: objective 1.650014e+09
Found heuristic solution: objective 1.616682e+09
Found heuristic solution: objective 1.600013e+09
Found heuristic solution: objective 1.600013e+09
Found heuristic solution: objective 1.600013e+09
Found heuristic solution: objective 1.500013e+09
Found heuristic solution: objective 1.500013e+09
Found heuristic solution: objective 1.475013e+09
Elapsed time for NoRel heuristic: 5s (best bound 8.00002e+08)

Root simplex log...

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    8.0000240e+08   1.843200e+04   0.000000e+00      5s
      72    8.0000240e+08   0.000000e+00   0.000000e+00      5s

Root relaxation: objective 8.000024e+08, 72 iterations, 0.00 seconds (0
```

NoRel Heuristic

Example: glass4

```
     Nodes    |    Current Node    |     Objective Bounds      |     Wor
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node

    0     0 8.0000e+08    0   72 1.4750e+09 8.0000e+08  45.8%     -
    0     0 8.0000e+08    0   72 1.4750e+09 8.0000e+08  45.8%     -
    0     2 8.0000e+08    0   72 1.4750e+09 8.0000e+08  45.8%     -
 12212  5296 1.1000e+09   47   46 1.4750e+09 8.0000e+08  45.8%   5.9
*12457  5028              66    1.400013e+09 8.0000e+08  42.9%   6.1
H12698  3108                    1.200013e+09 8.0000e+08  33.3%   6.2

Explored 17122 nodes (115780 simplex iterations) in 11.99 seconds (6.38
Thread count was 8 (of 8 available processors)

Solution count 10: 1.20001e+09 1.40001e+09 1.47501e+09 ... 1.65001e+09

Optimal solution found (tolerance 1.00e-04)
Best objective 1.200012600000e+09, best bound 1.200003200000e+09, gap 0
```

Custom Heuristics
When?

Custom Heuristics
When?

- Solver cannot find any solution

Custom Heuristics
When?

- Solver cannot find any solution
- Solution quality is not sufficient

Custom Heuristics
When?

- Solver cannot find any solution
- Solution quality is not sufficient
- Solution improvement too slow

Custom Heuristics
When?

- Solver cannot find any solution
- Solution quality is not sufficient
- Solution improvement too slow
- Model is too large

Custom Heuristics
What can we do?

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial
  - No need for high solution quality

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial
  - No need for high solution quality
- **Improve solutions in callbacks**:

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial
  - No need for high solution quality
- **Improve solutions in callbacks**:
  - Round LP solutions with problem-specific knowledge

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial
  - No need for high solution quality
- **Improve solutions in callbacks**:
  - Round LP solutions with problem-specific knowledge
  - Improve integer solutions with local-search techniques

Custom Heuristics
What can we do?

- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial
  - No need for high solution quality
- **Improve solutions in callbacks**:
  - Round LP solutions with problem-specific knowledge
  - Improve integer solutions with local-search techniques
- **Partition model** by time, location, etc.

Custom Heuristics
What can we do?

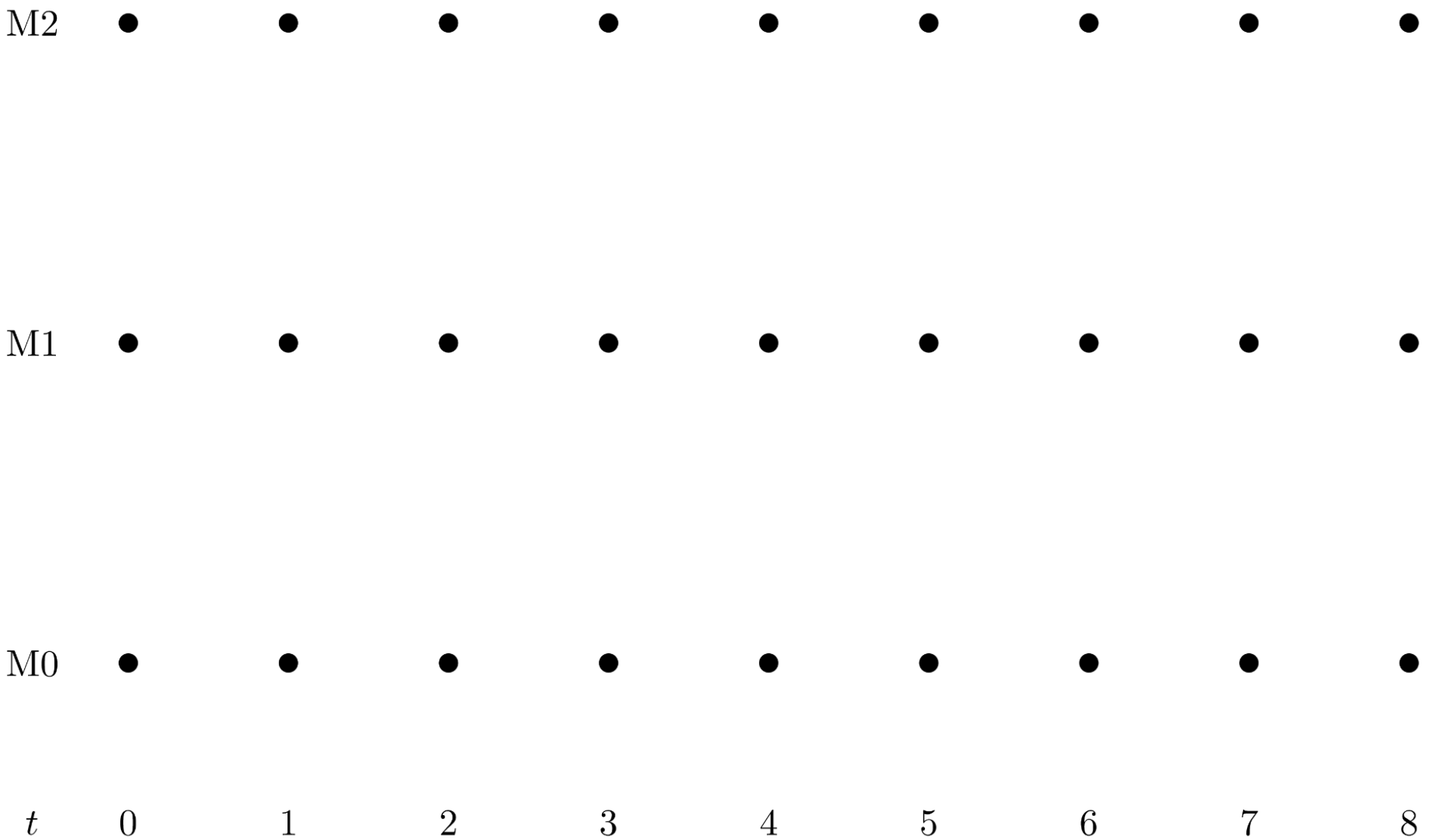
- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial
  - No need for high solution quality
- **Improve solutions in callbacks**:
  - Round LP solutions with problem-specific knowledge
  - Improve integer solutions with local-search techniques
- **Partition model** by time, location, etc.
  - Variable attribute `Partition`, parameter `PartitionPlace`

Custom Heuristics
What can we do?

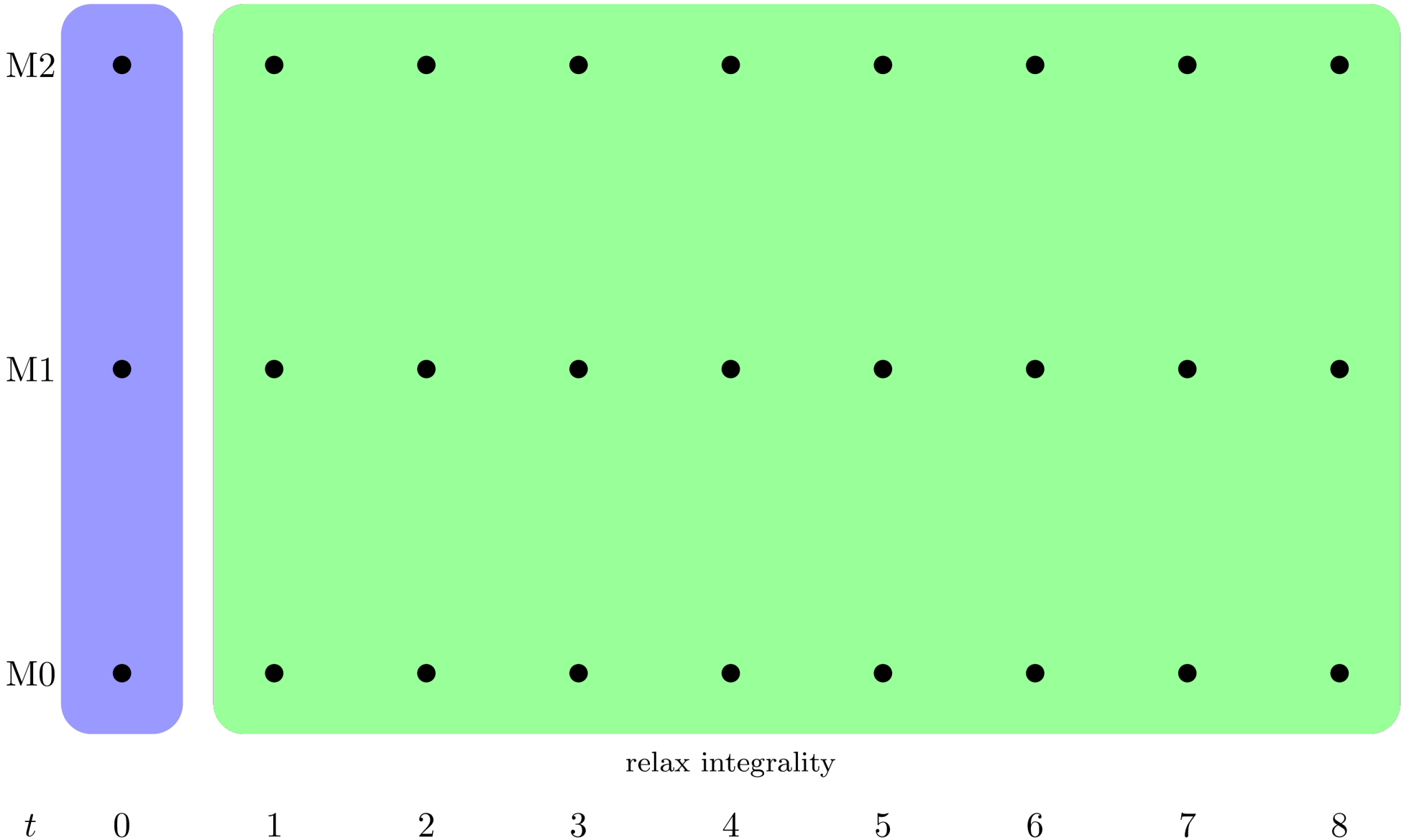
- **Make model easier to be feasible**:
  - Relax constraints
  - Use penalties for hard constraints
- **Construct initial solution** and hand it over as MIP start:
  - Can be complete or partial
  - No need for high solution quality
- **Improve solutions in callbacks**:
  - Round LP solutions with problem-specific knowledge
  - Improve integer solutions with local-search techniques
- **Partition model** by time, location, etc.
  - Variable attribute `Partition`, parameter `PartitionPlace`
  - Guides solver heuristics

Example: Rolling Horizon MIP Heuristic

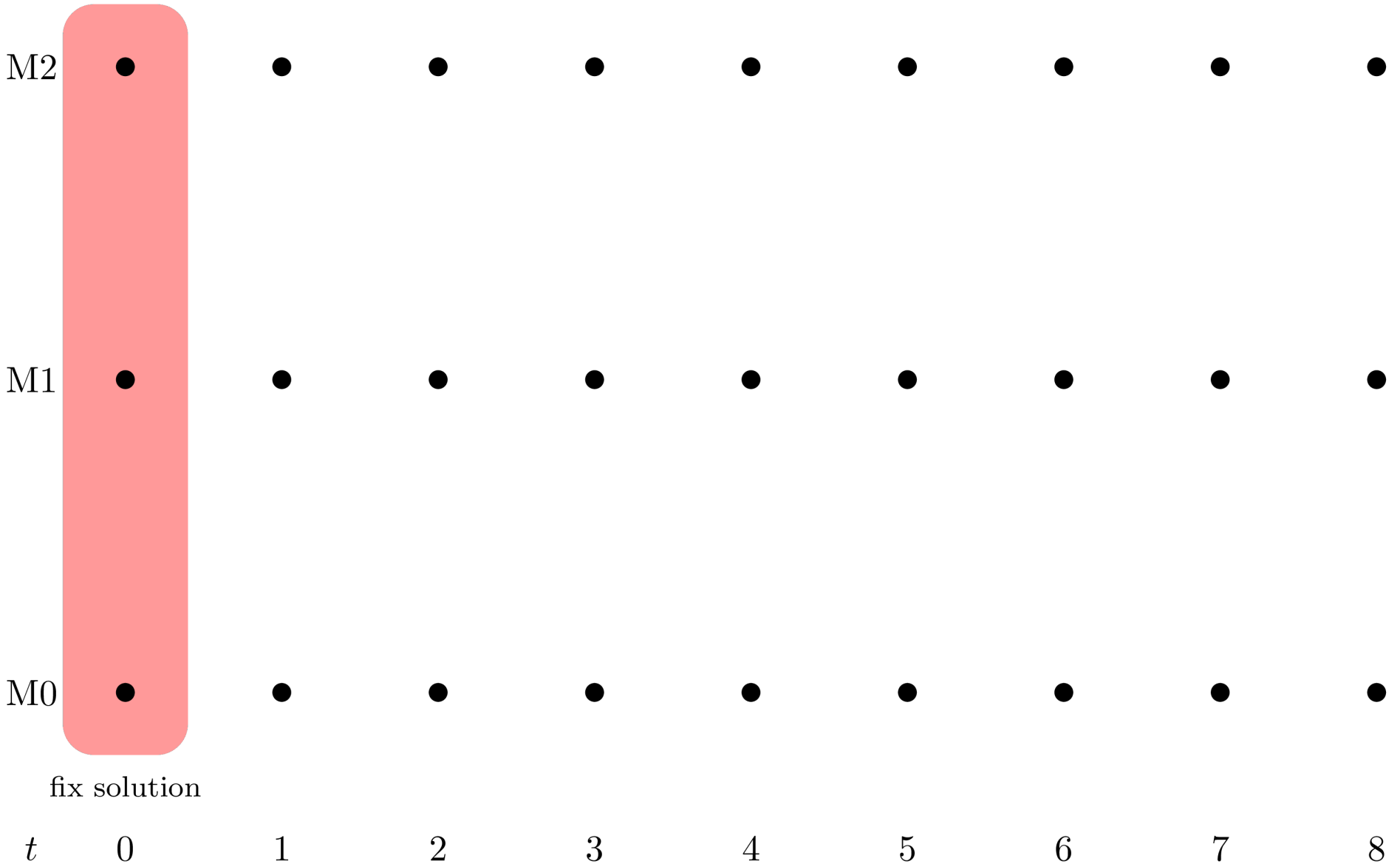
Example: Rolling Horizon MIP Heuristic



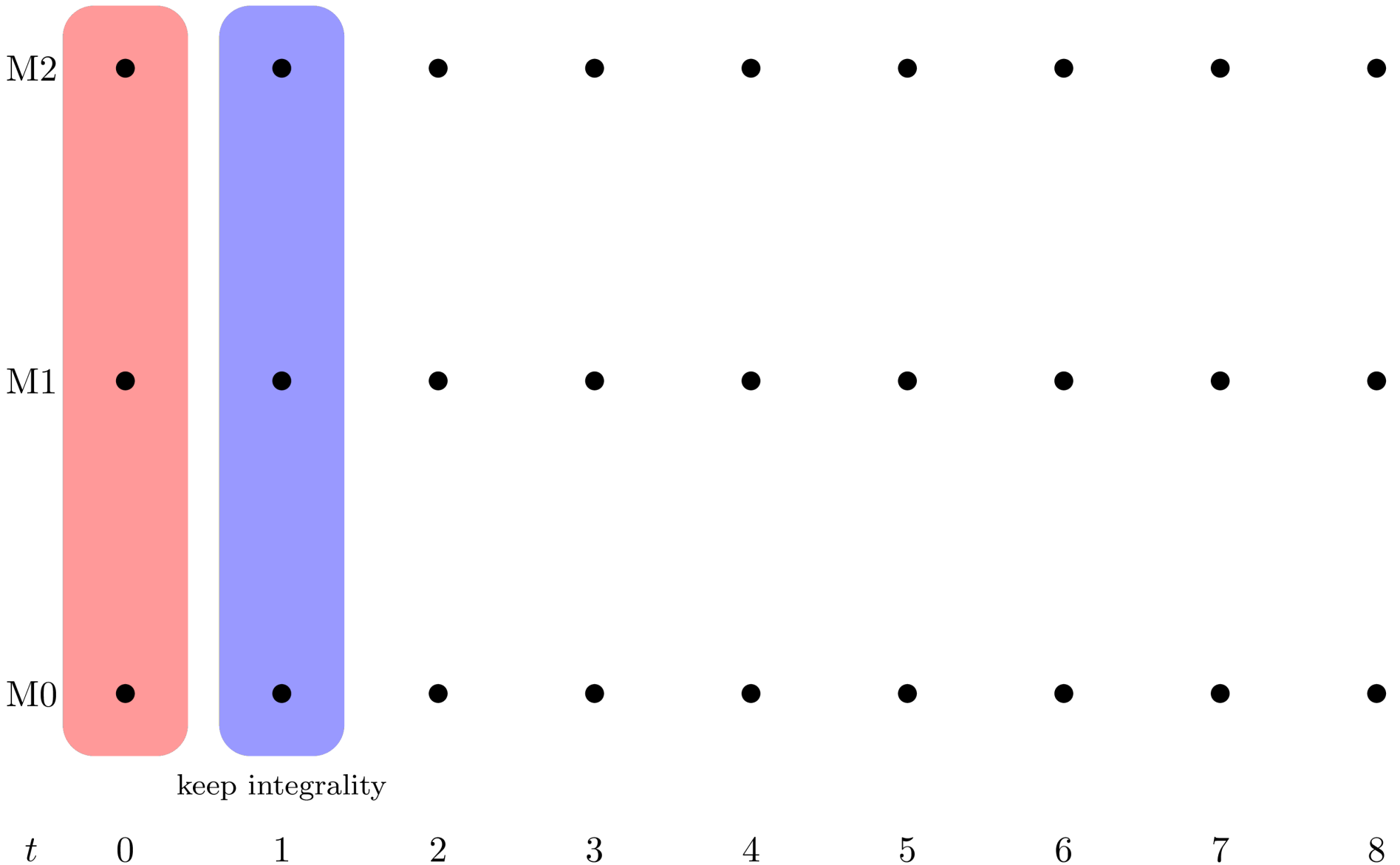
Example: Rolling Horizon MIP Heuristic



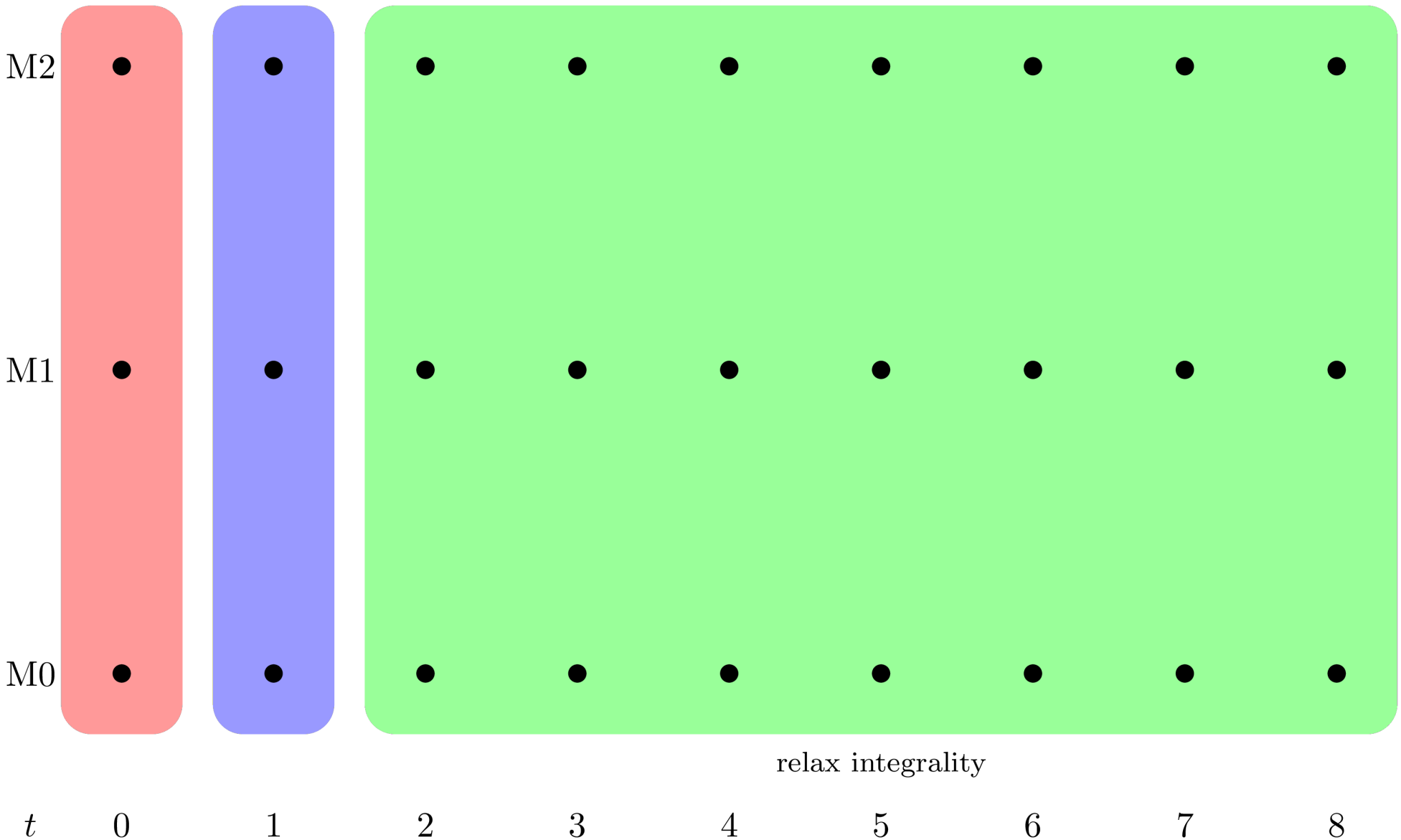
Example: Rolling Horizon MIP Heuristic



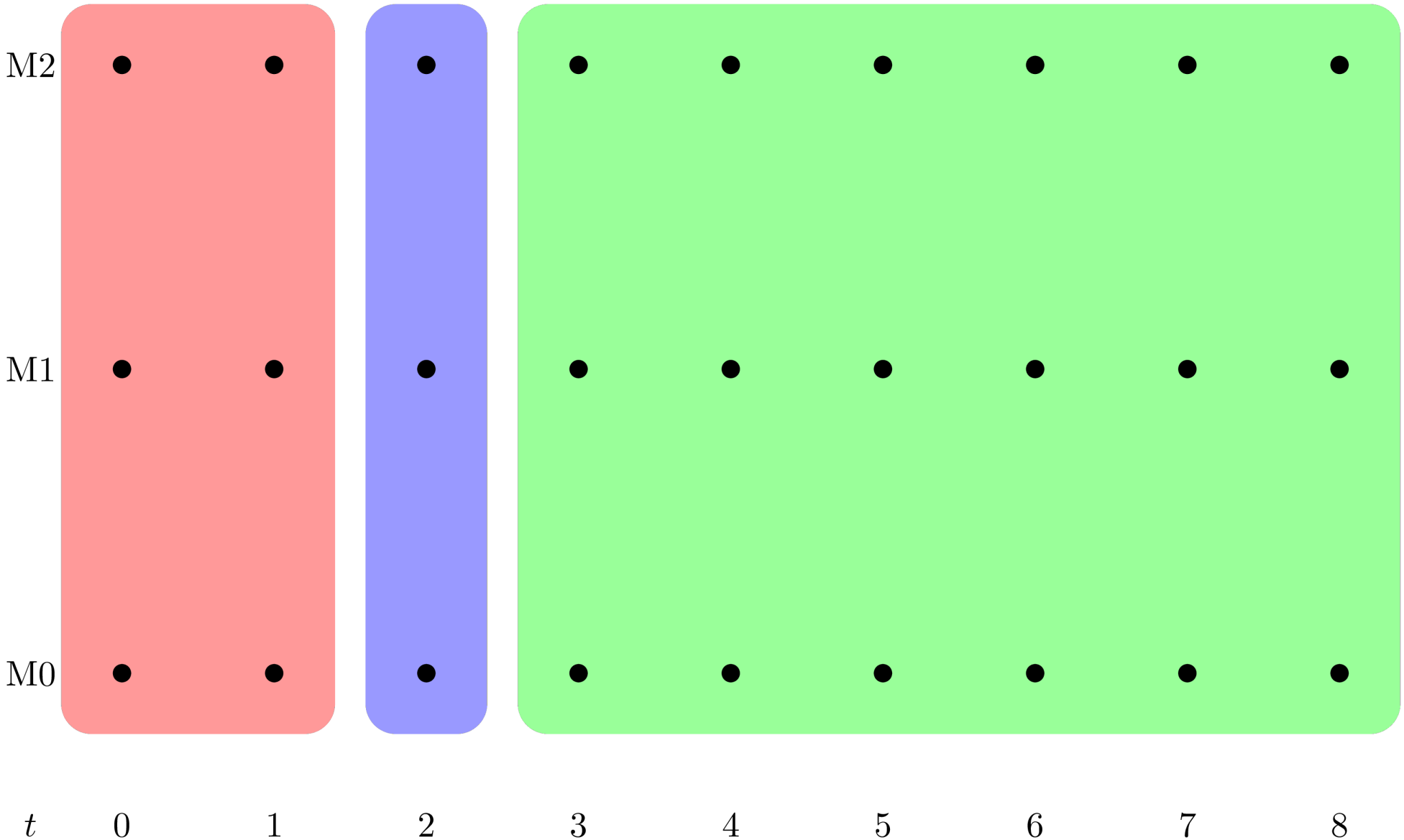
Example: Rolling Horizon MIP Heuristic



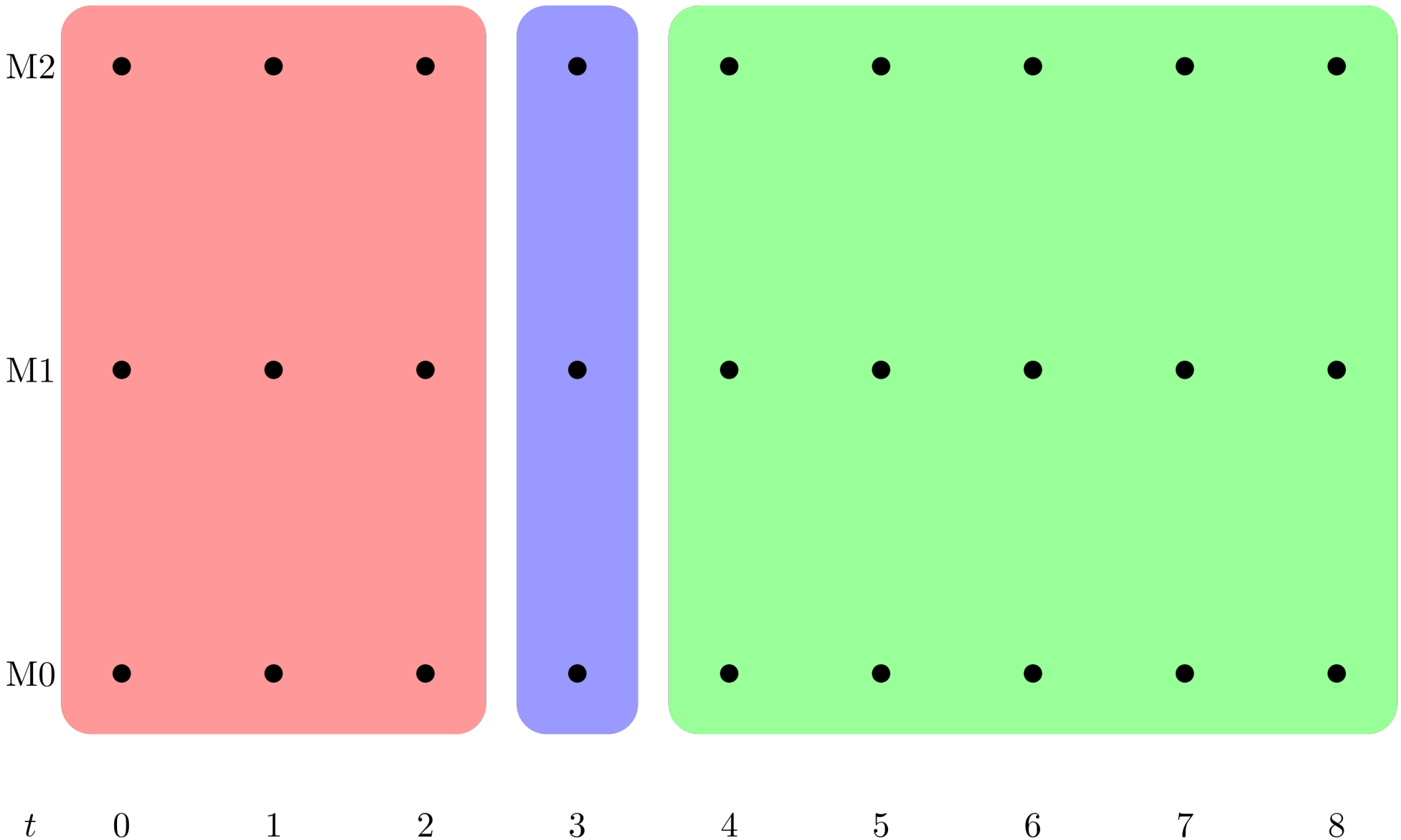
Example: Rolling Horizon MIP Heuristic



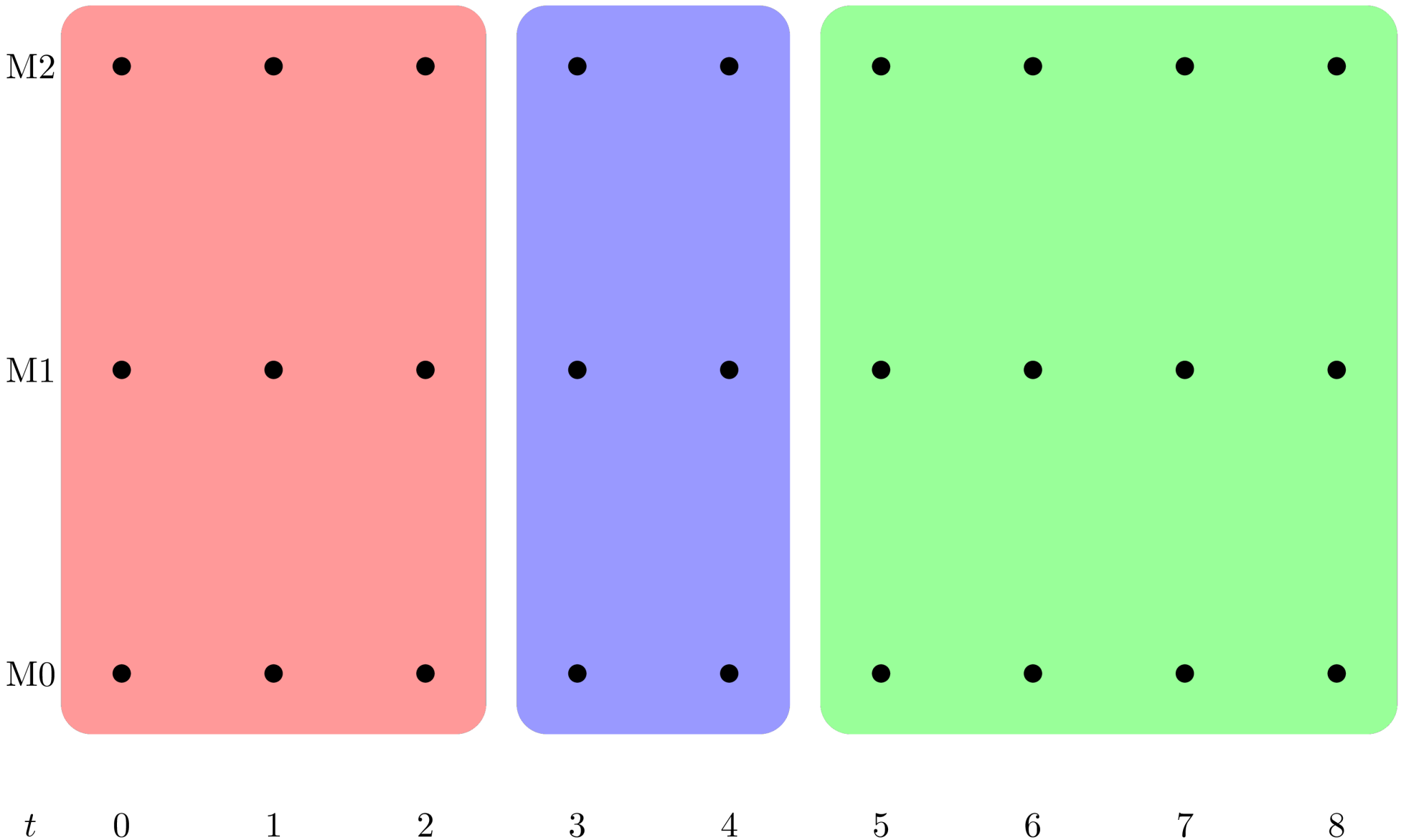
Example: Rolling Horizon MIP Heuristic



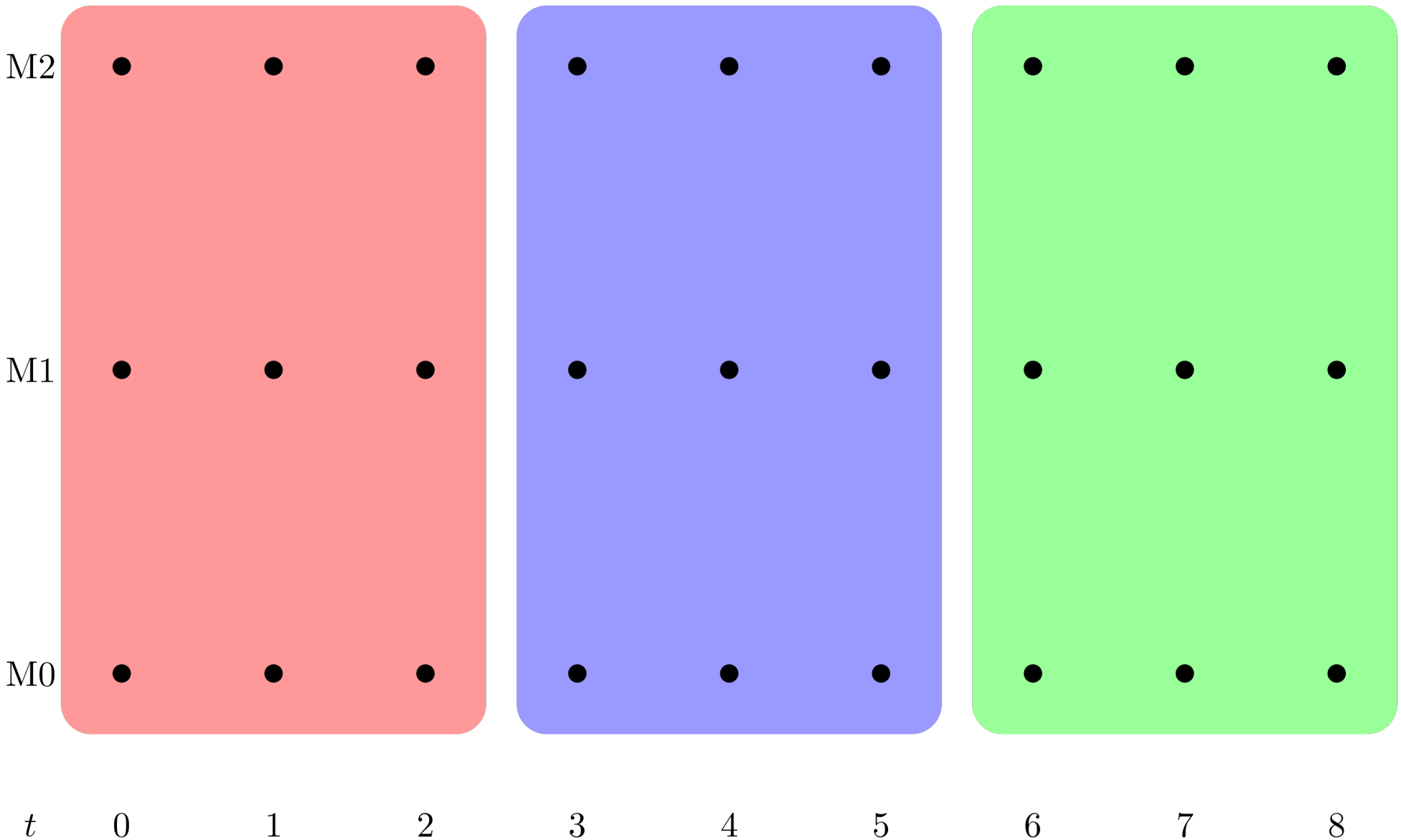
Example: Rolling Horizon MIP Heuristic



Example: Rolling Horizon MIP Heuristic

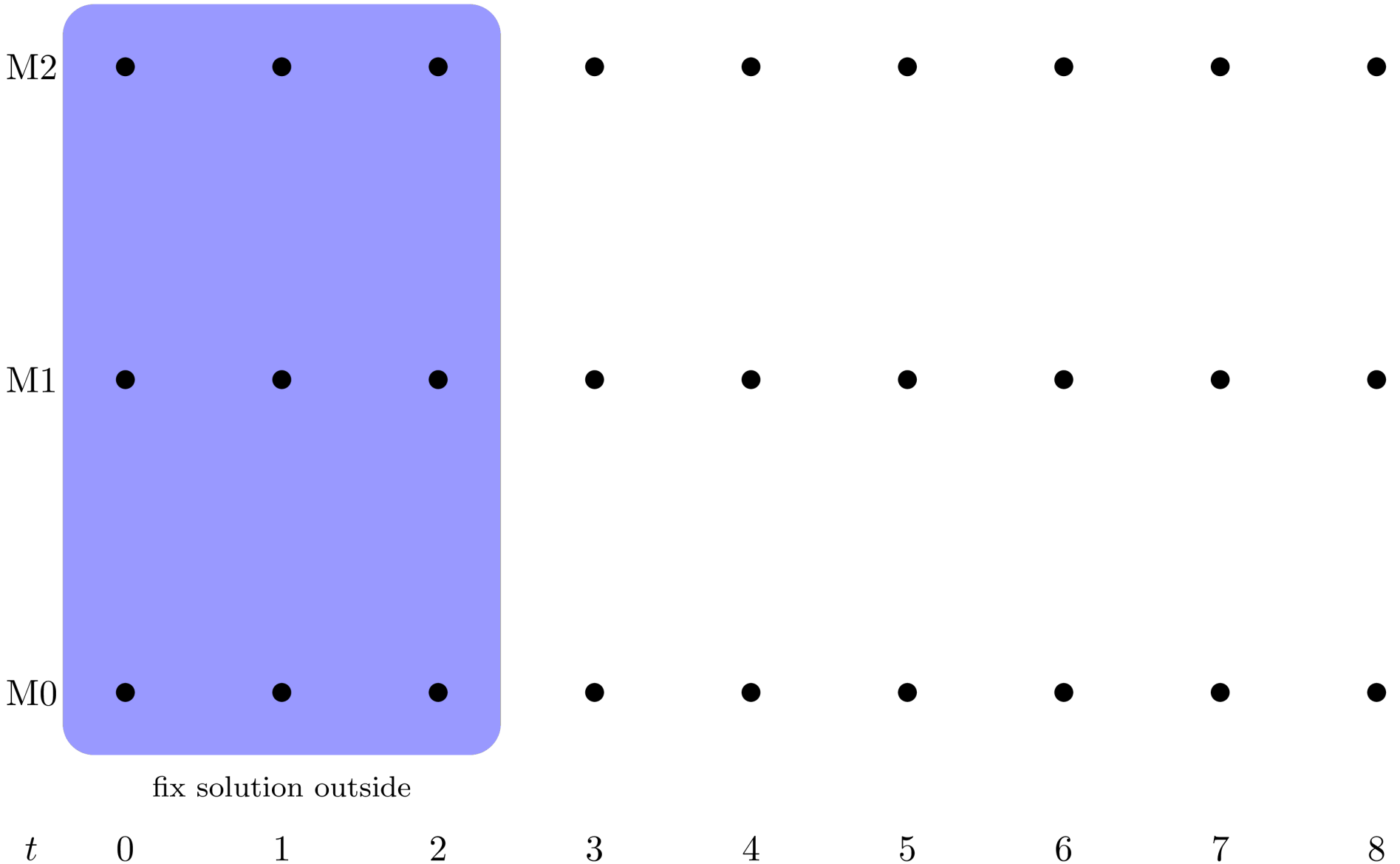


Example: Rolling Horizon MIP Heuristic

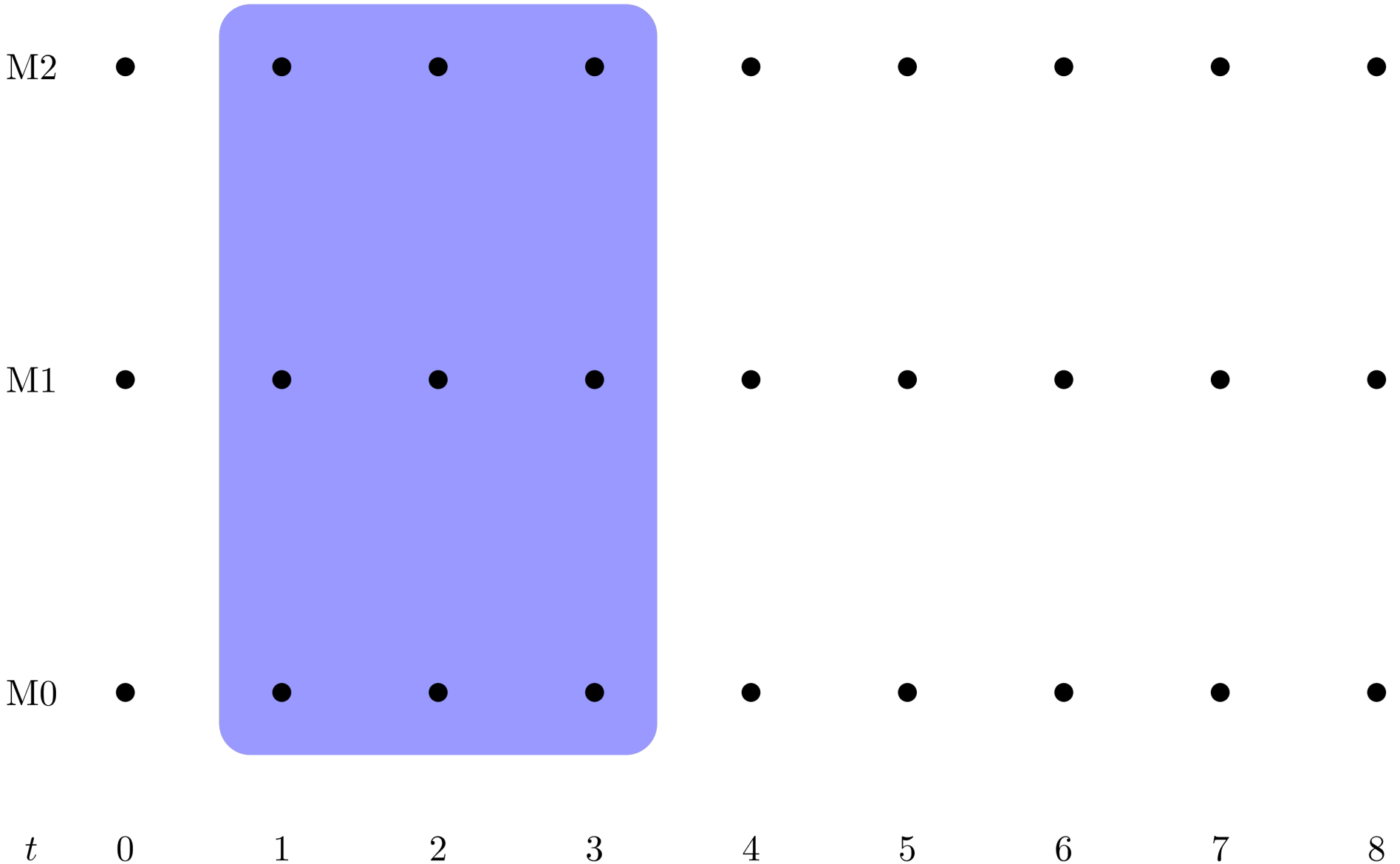


Example: Local Search MIP Heuristic

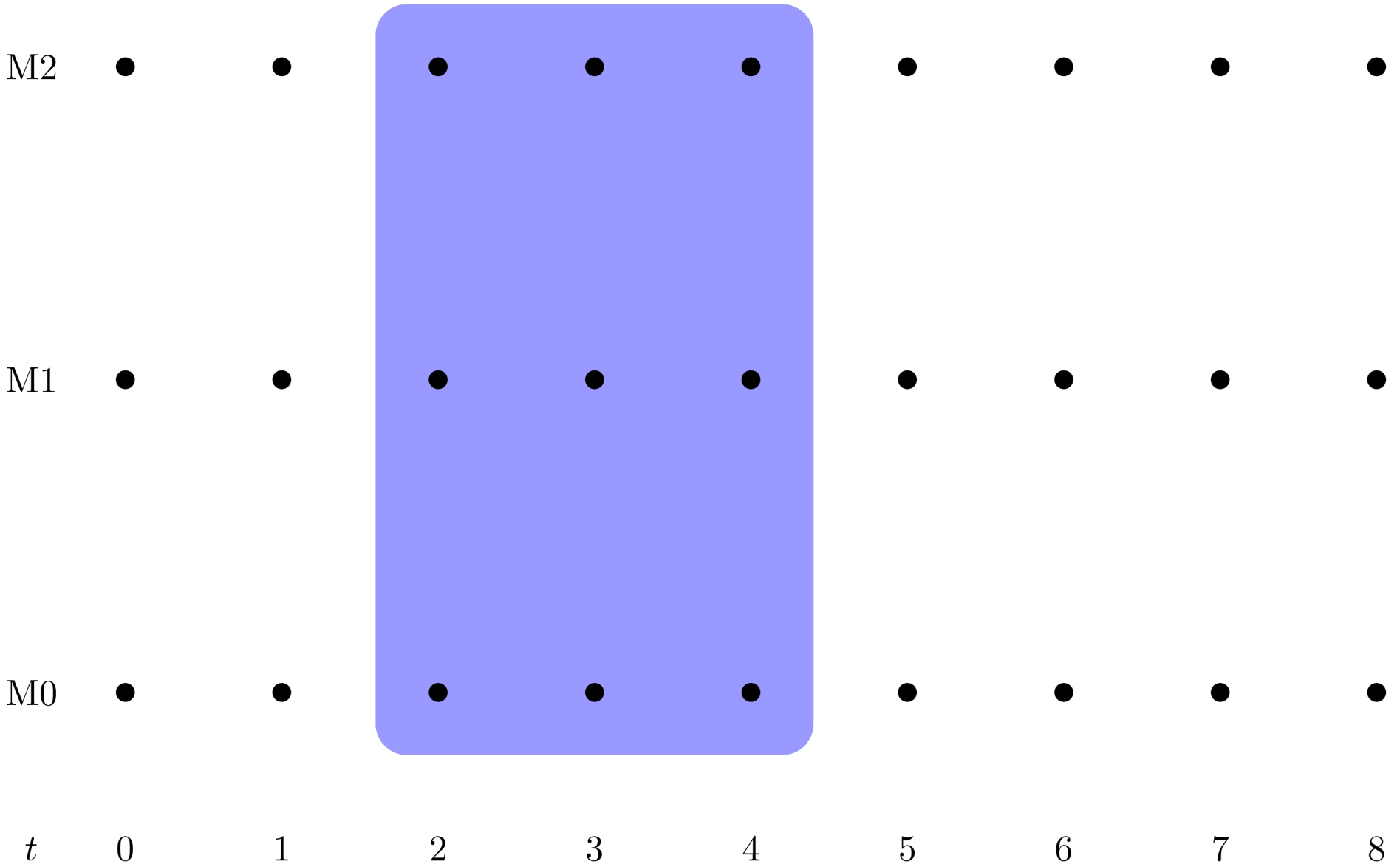
Example: Local Search MIP Heuristic



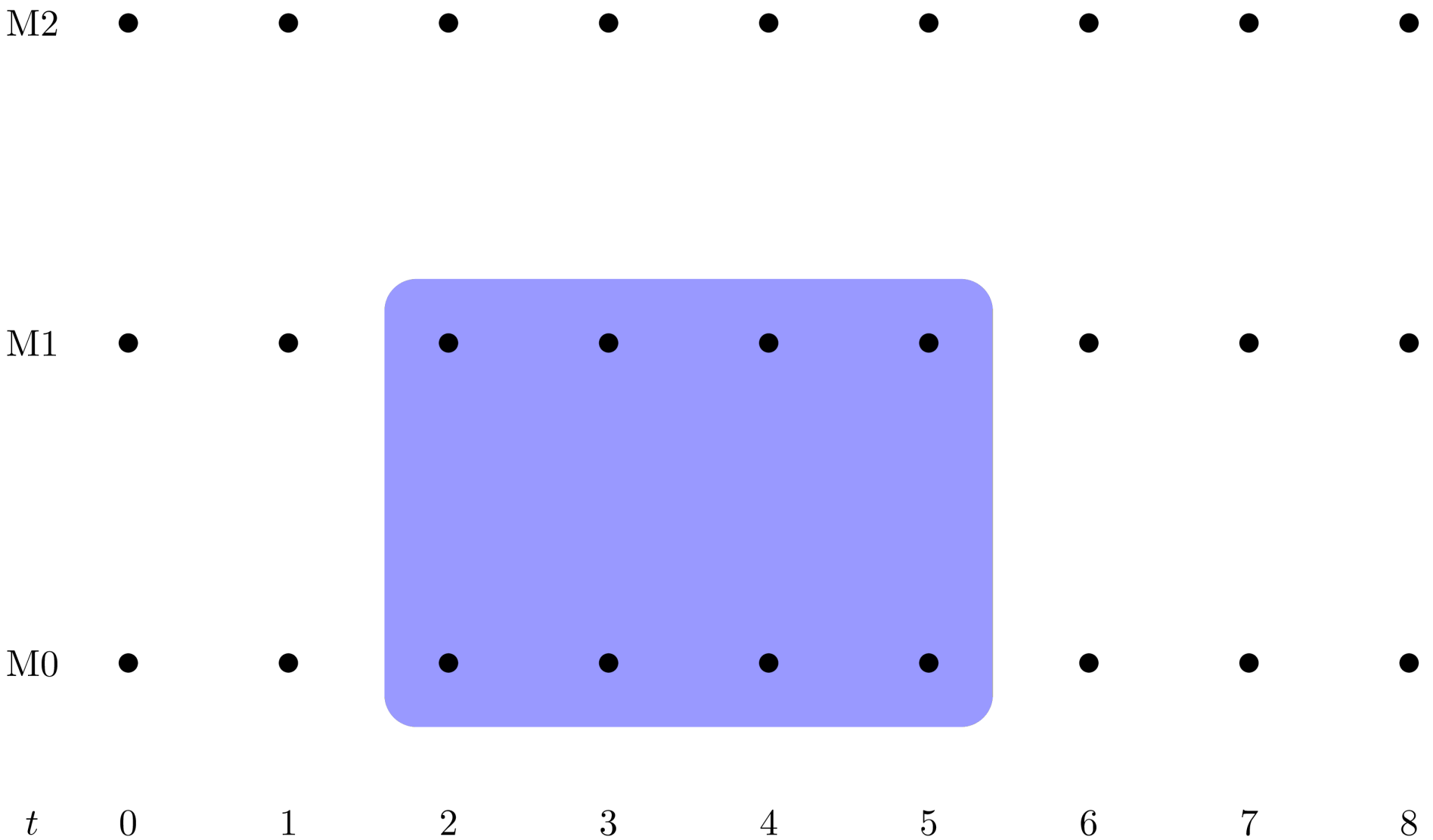
Example: Local Search MIP Heuristic



Example: Local Search MIP Heuristic



Example: Local Search MIP Heuristic



Conclusion
How to handle challenging models?

Conclusion
How to handle challenging models?

- **Modeling phase**

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values
    - Add problem-specific cuts

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values
    - Add problem-specific cuts
  - Reformulate model

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values
    - Add problem-specific cuts
  - Reformulate model
- **Solution phase**

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values
    - Add problem-specific cuts
  - Reformulate model
- **Solution phase**
  - Analyze solver logs → Gurobi Experts

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values
    - Add problem-specific cuts
  - Reformulate model
- **Solution phase**
  - Analyze solver logs → Gurobi Experts
  - Run custom heuristics

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values
    - Add problem-specific cuts
  - Reformulate model
- **Solution phase**
  - Analyze solver logs → Gurobi Experts
  - Run custom heuristics
  - Analyze model structure → Gurobi Experts

Conclusion
How to handle challenging models?

- **Modeling phase**
  - Clean up input data
  - Understand model strength
    - Tighten variable bounds
    - Reduce Big-M values
    - Add problem-specific cuts
  - Reformulate model
- **Solution phase**
  - Analyze solver logs → Gurobi Experts
  - Run custom heuristics
  - Analyze model structure → Gurobi Experts
  - Tune solver parameters → Gurobi Experts