Linear Programming

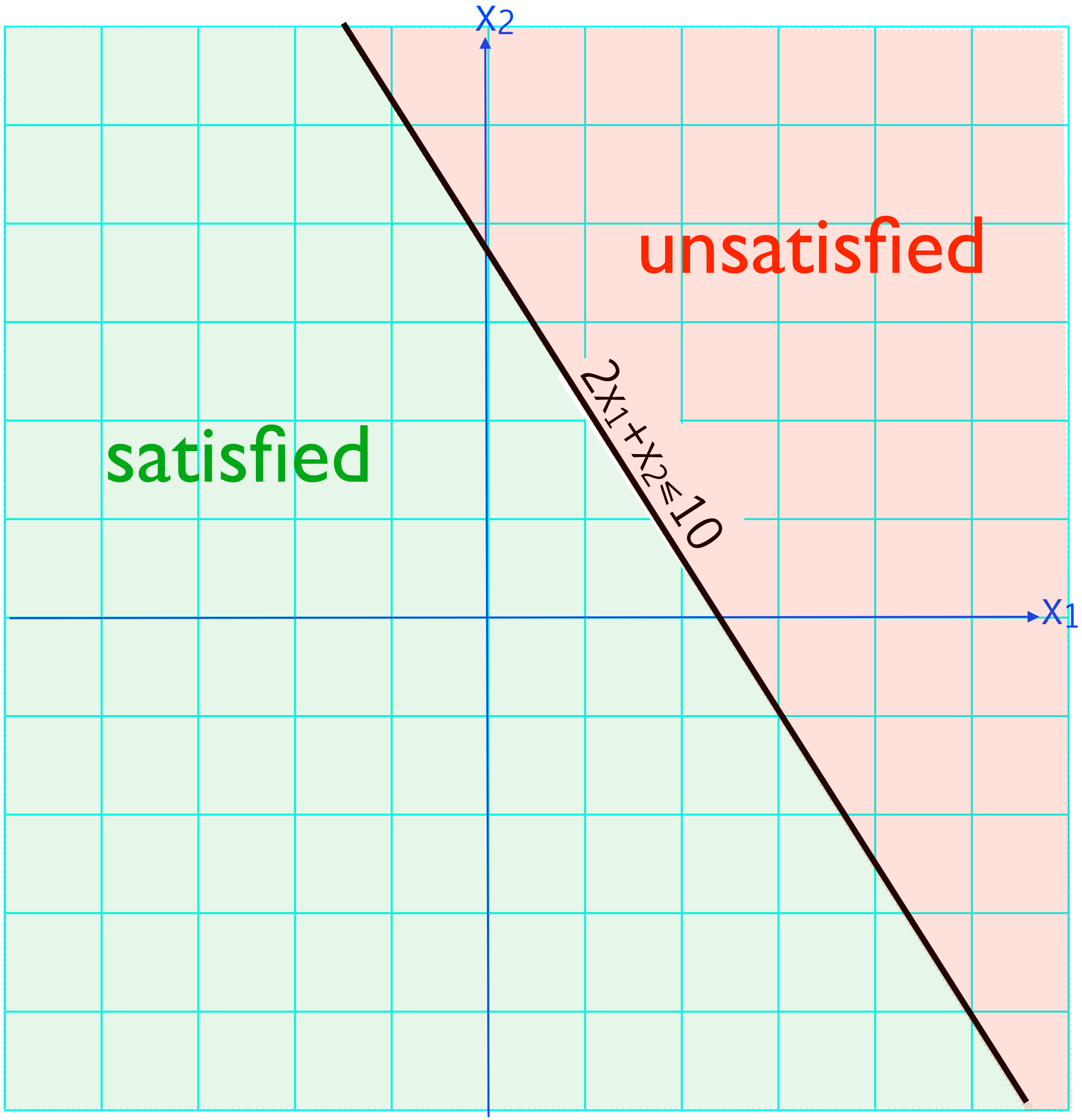
Linear Programs - example 1

$$\begin{array}{llcrcr} \text{maximize} & x_1 & + & x_2 & & \\ \text{subject to} & & & & & \\ & 4x_1 & - & x_2 & \leq & 8 \\ & 2x_1 & + & x_2 & \leq & 10 \\ & 5x_1 & - & 2x_2 & \geq & -2 \\ & & x_1, x_2 & & \geq & 0 \, . \end{array}$$

- Optimization problem
- x_1, x_2 = variables
- $z = x_1 + x_2$ = objective
  - linear in x variables
- "subject to" constraints
  - $4x_1 - x_2 \leq 8$
  - $2x_1 + x_2 \leq 10$
  - $5x_1 - 2x_2 \geq -2$
  - $x_1, x_2 \geq 0$
  - also linear in x variables

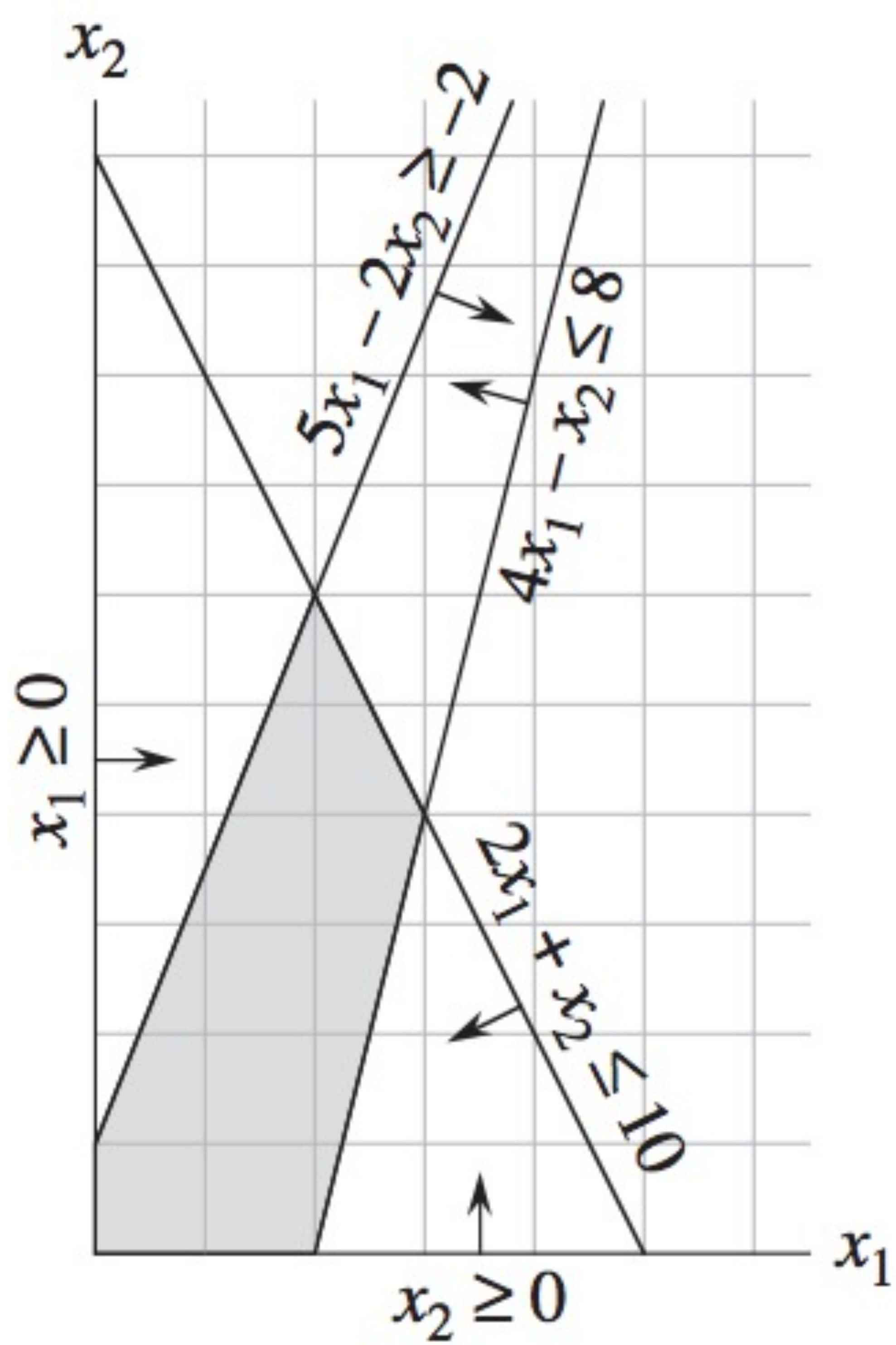
Linear programs – feasible region

- Each linear constraint "splits" the space into two halves
  - "satisfied" half (constraint holds)
  - "unsatisfied" half (constraint doesnt hold)
  - separation is a line given by the constraint



Linear programs – feasible region

- Feasible region = intersection of "satisfied" halfs for all constraints
- clearly solution(s) (x1,x2) must be in this feasible region
  - any other (x1,x2) outside this region violates some constraint(s)

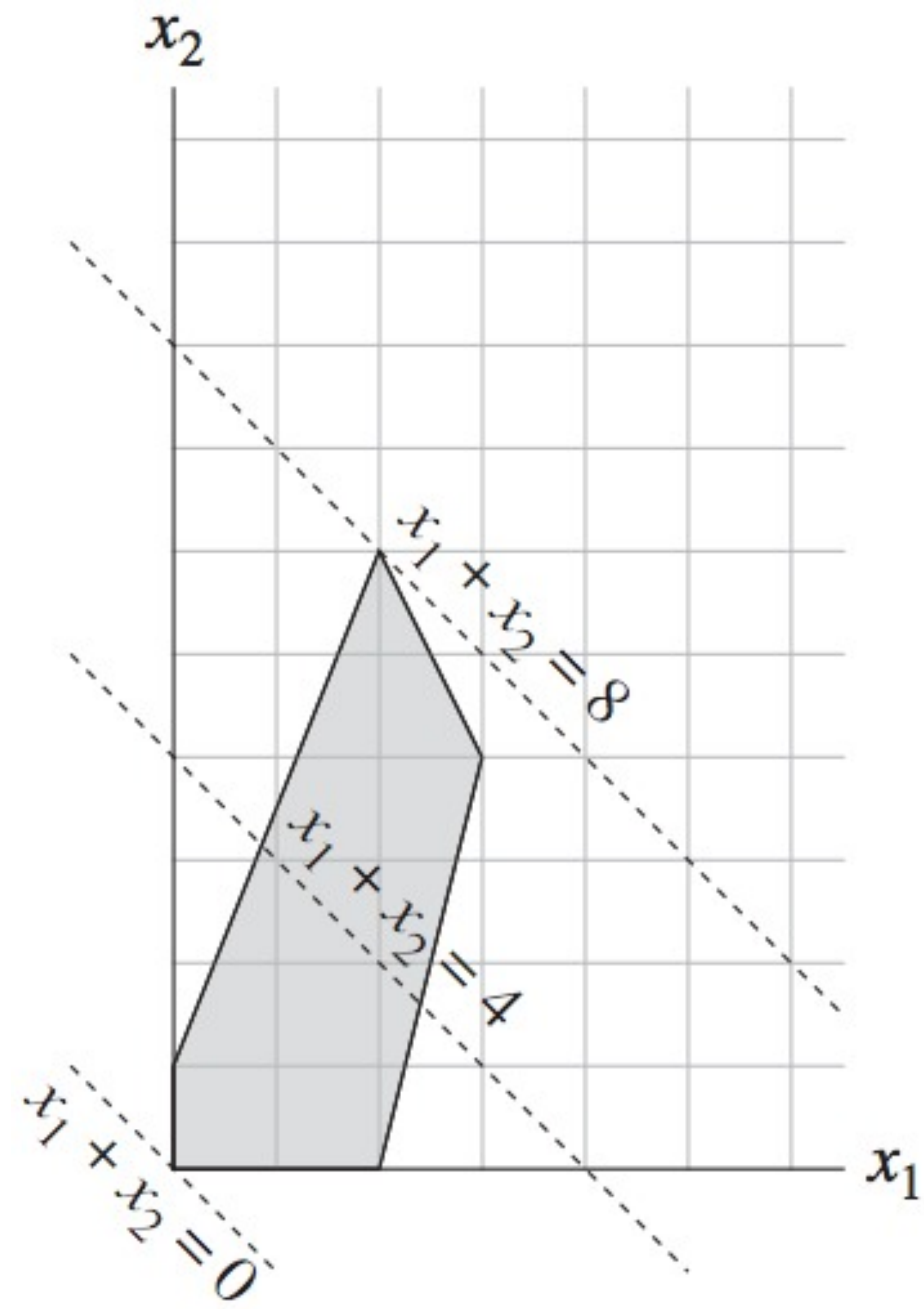


Linear Programs - Objective

- z = x1+x2 is objective, to be maximized (want the max z)
  - other times want the min, "minimized"

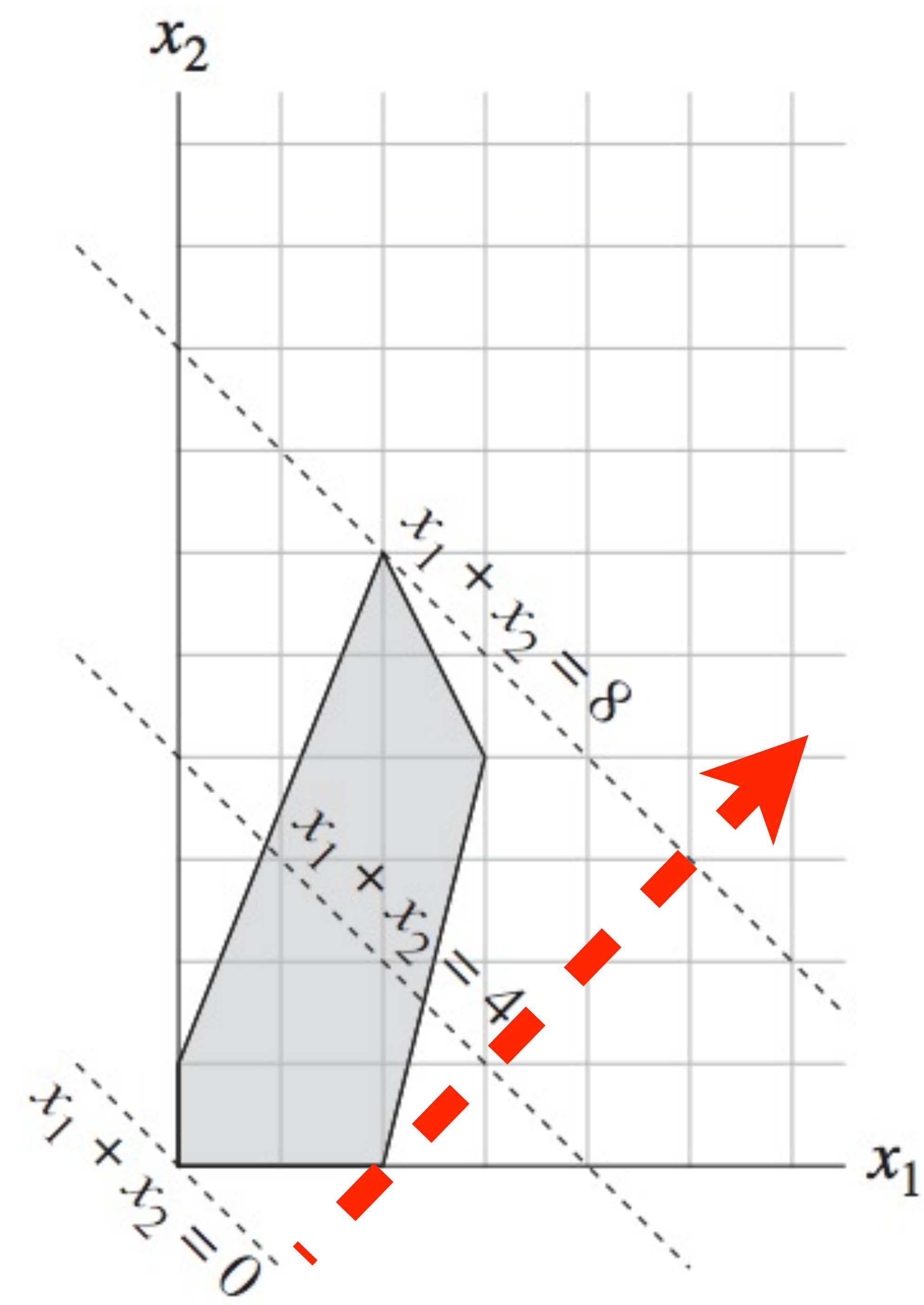
Linear Programs – Objective

- z = x1+x2 is objective, to be maximized (want the max z)
  - other times want the min, "minimized"
- for a fixed z, z=x1+x2 is a line
  - "z line" or "objective line"
  - 3 z lines drawn for z=0, z=4, z=8
  - on each such line, any (x1,x2) gives in the same objective



Linear Programs - Objective

- z = x1+x2 is objective, to be maximized (want the max z)
  - other times want the min, "minimized"
- for a fixed z, z=x1+x2 is a line
  - "z line" or "objective line"
  - 3 z lines drawn for z=0, z=4, z=8
  - on each such line, any (x1,x2) gives in the same objective
- only interested in y objective lines that intersect the feasible region
  - out of these we want the "last" line that intersects FR, in the direction of max objective (dotted red direction)
  - the last intersection objective line is y=8



Linear Programs - example 2

maximize

$$z = 3x_1 + 5x_2$$

subject to

$$x_1 \leq 4$$

$$2x_2 \leq 12$$

$$3x_1 + 2x_2 \leq 18$$

$$x_1, x_2 \geq 0$$

Linear Programs - example 2

maximize

$$z = 3x_1 + 5x_2$$

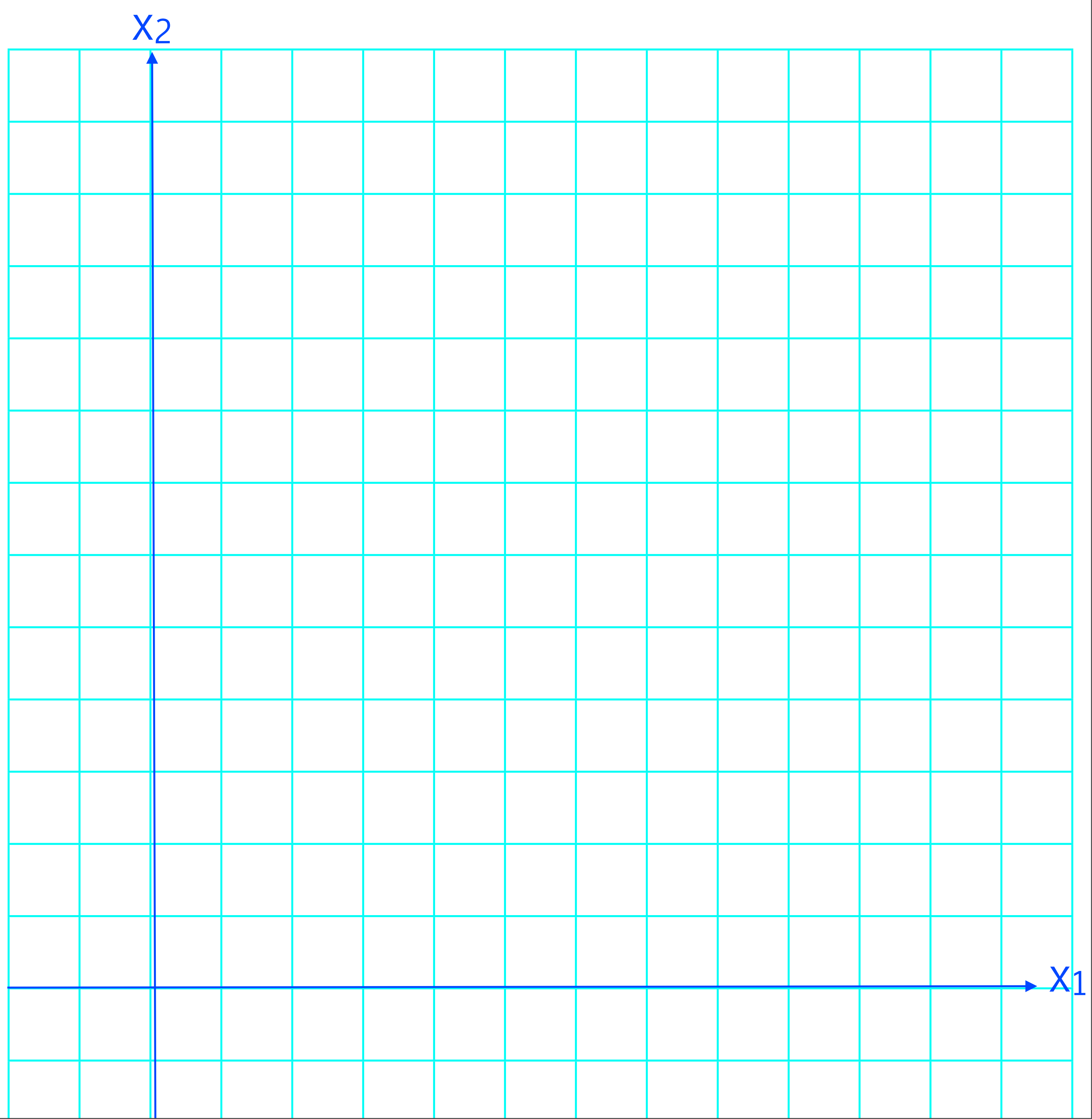
subject to

$$x_1 \le 4$$

$$2x_2 \le 12$$

$$3x_1 + 2x_2 \le 18$$

$$x_1, x_2 \ge 0$$



Linear Programs – example 2

maximize

$$z = 3x_1 + 5x_2$$

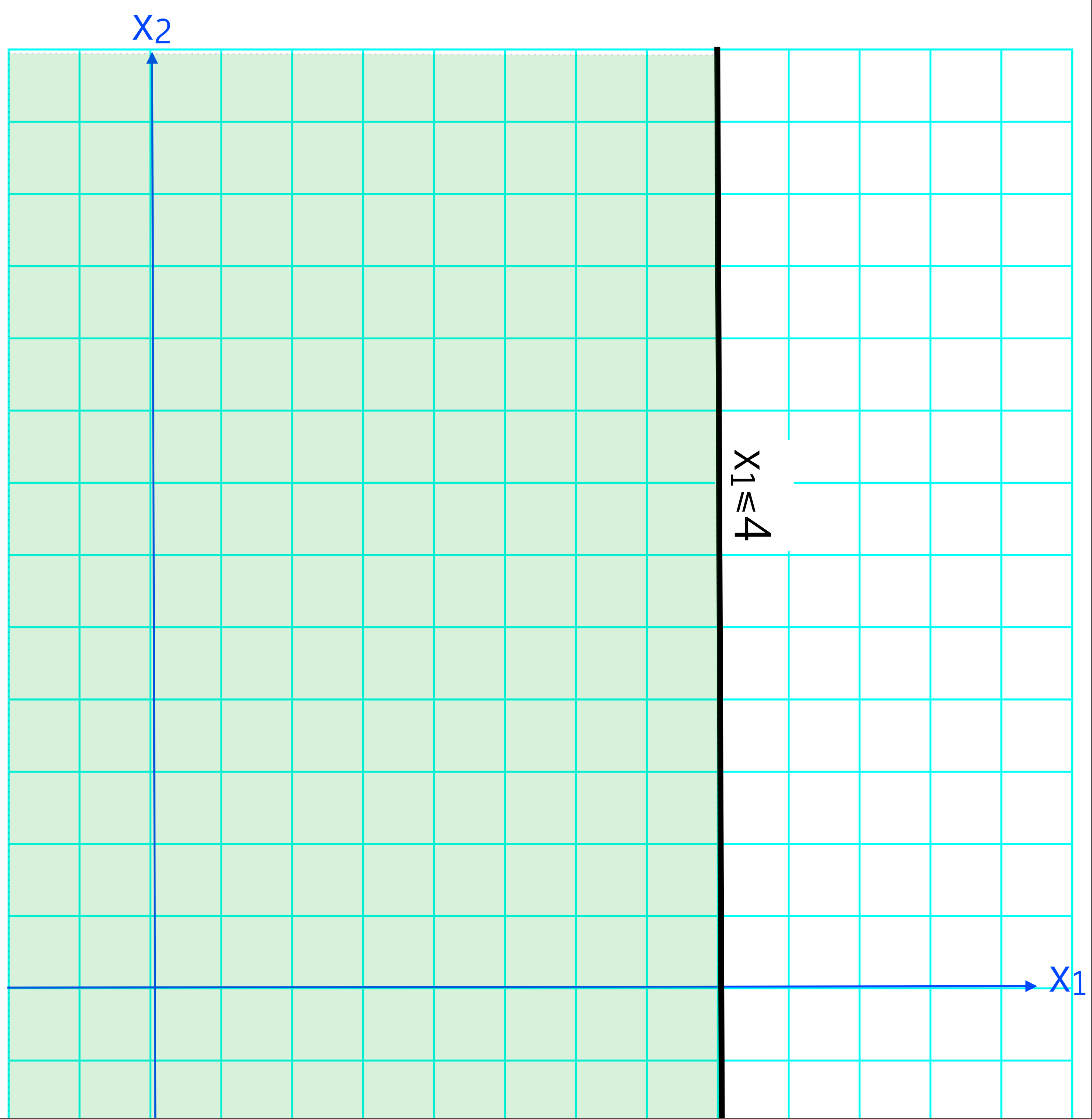
subject to

$$x_1 \le 4$$

$$2x_2 \le 12$$

$$3x_1 + 2x_2 \le 18$$

$$x_1, x_2 \ge 0$$



Linear Programs – example 2

maximize

$$z = 3x_1 + 5x_2$$

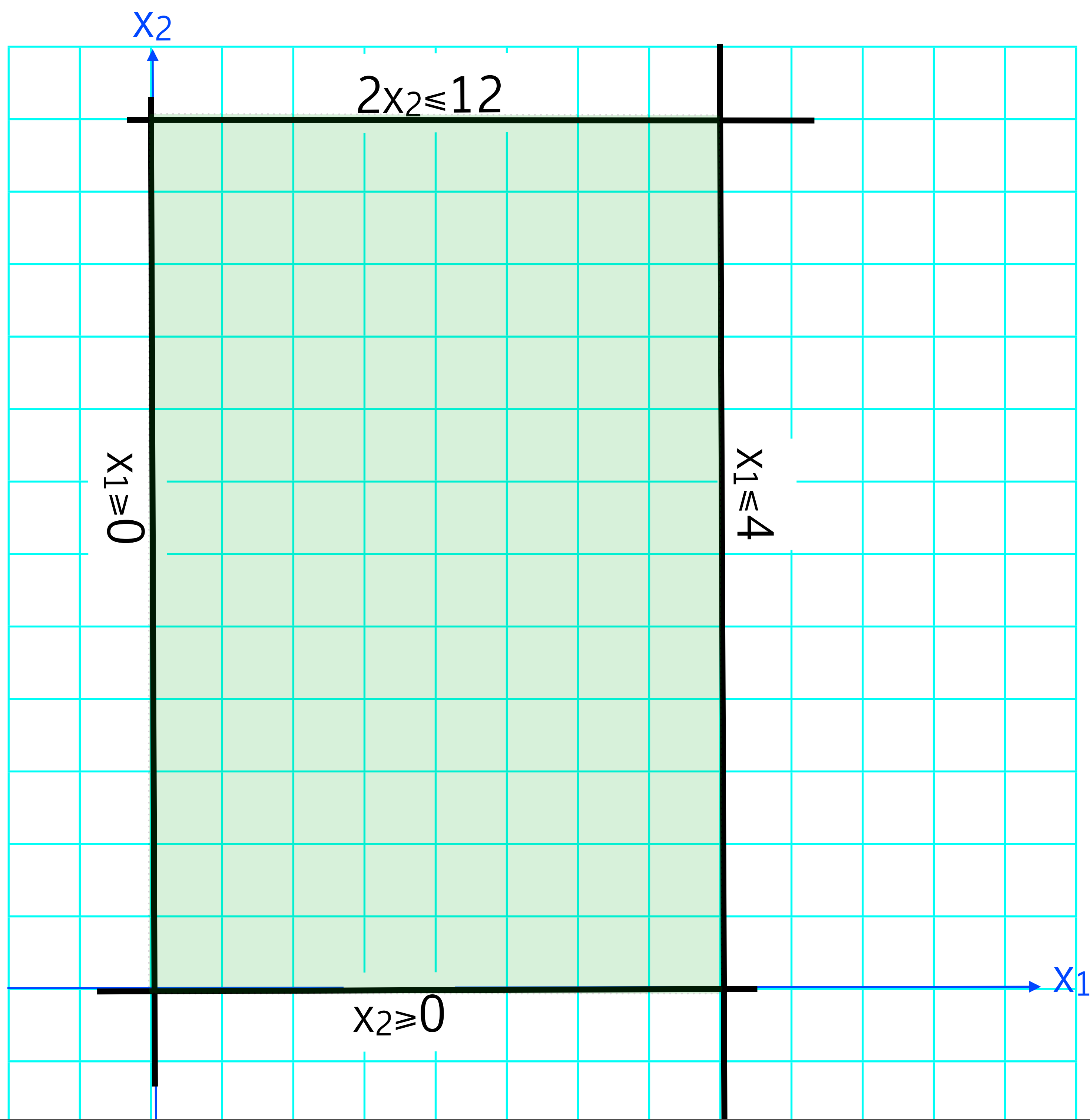
subject to

$$x_1 \leq 4$$

$$2x_2 \leq 12$$

$$3x_1 + 2x_2 \leq 18$$

$$x_1, x_2 \geq 0$$



Linear Programs - example 2

maximize

$$z = 3x_1 + 5x_2$$

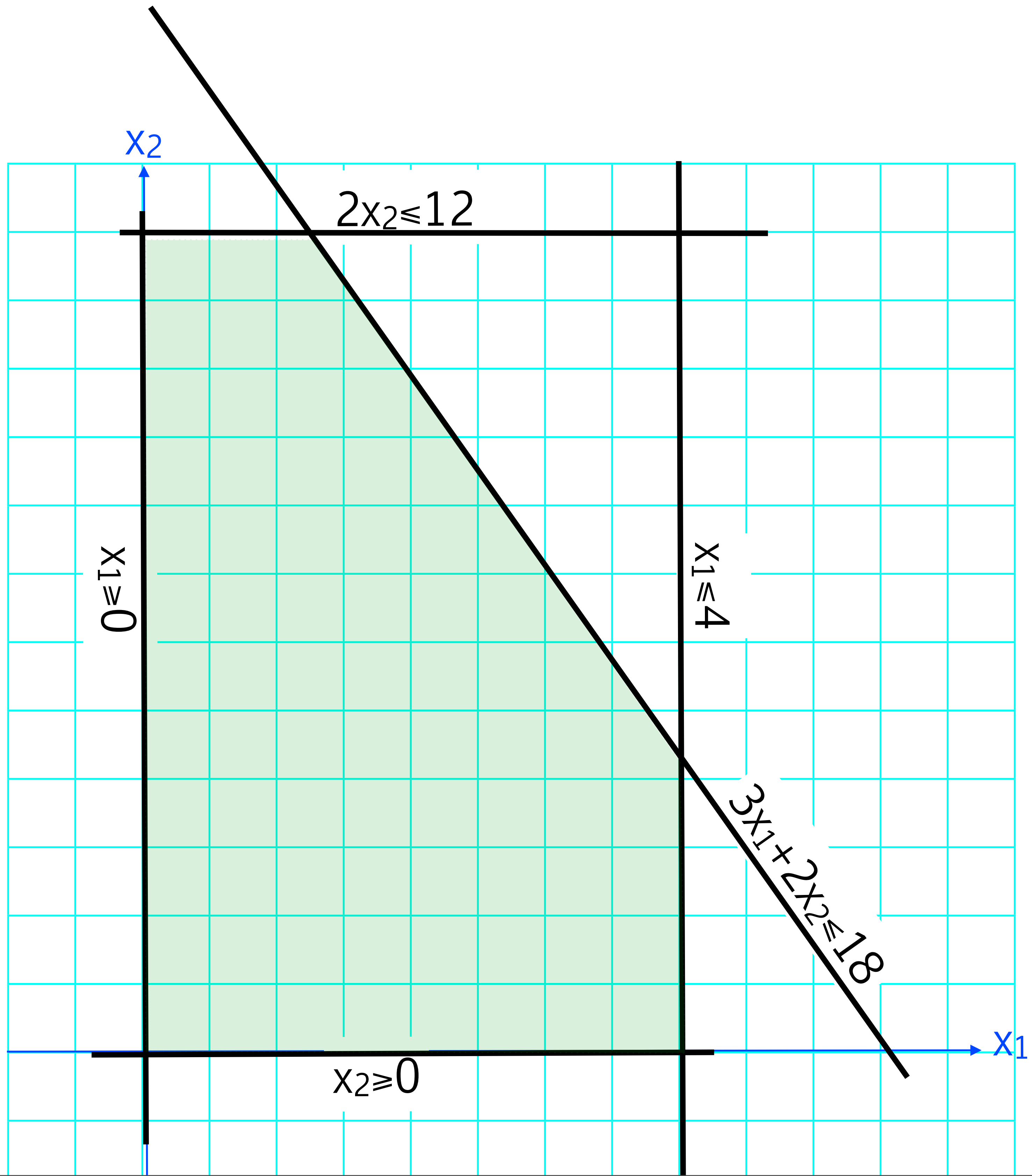
subject to

$$x_1 \leq 4$$

$$2x_2 \leq 12$$

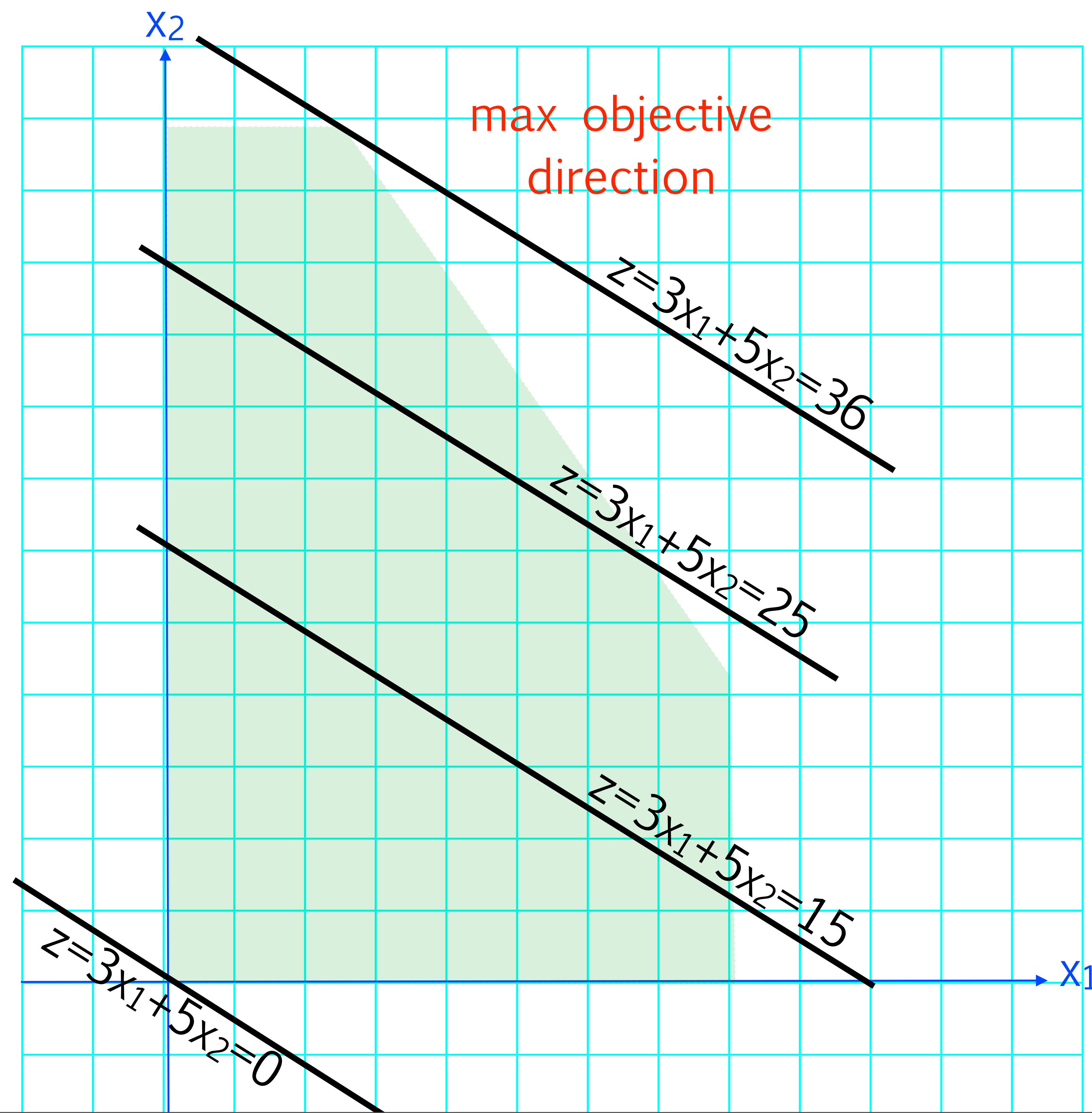
$$3x_1 + 2x_2 \leq 18$$

$$x_1, x_2 \geq 0$$



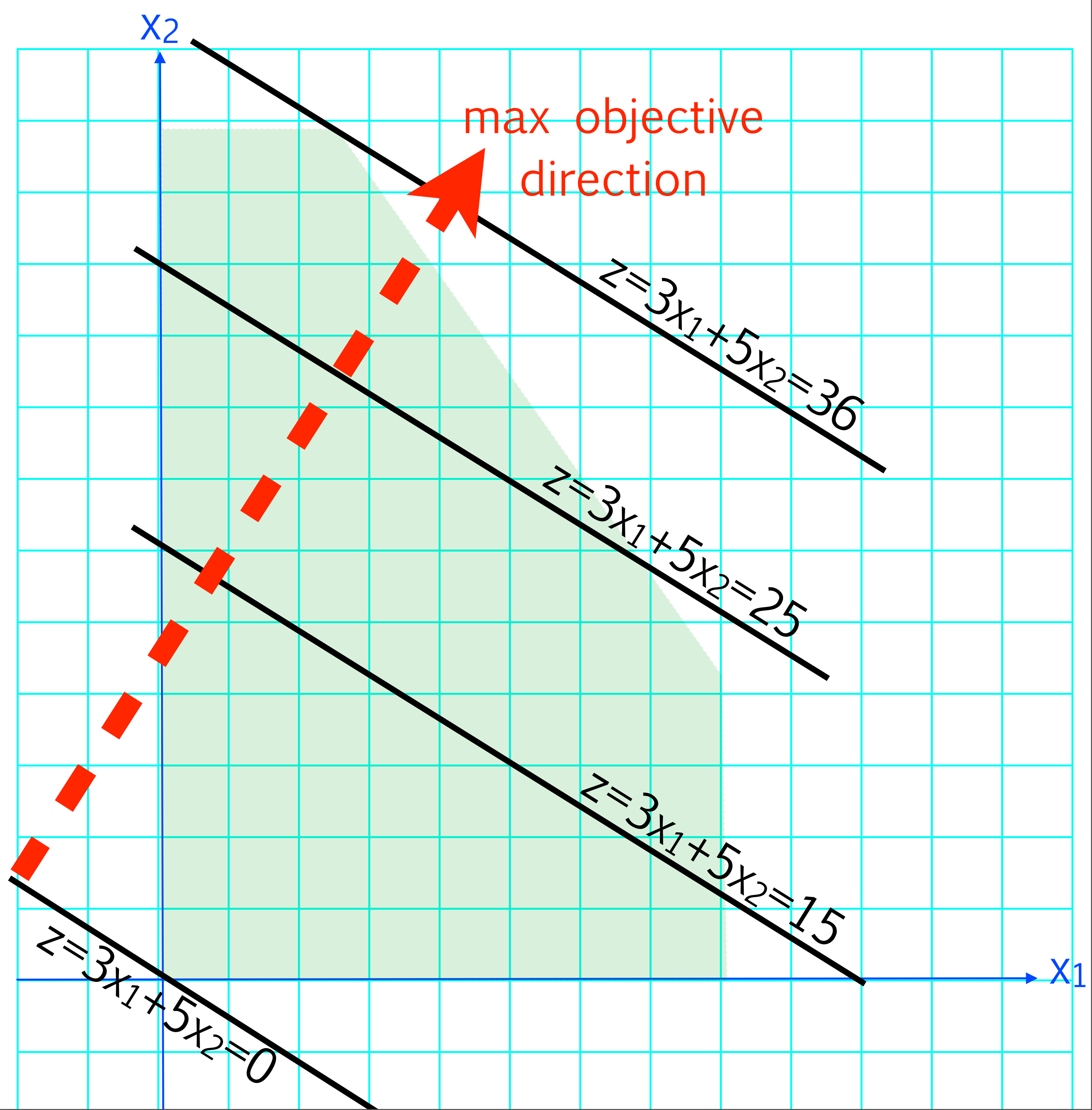
Linear Programs - example 2

- objective
  $z=3x_1+5x_2$
  - 4 objective lines drawn: z=0,15,25,36
- last z line intersecting feasible reagion: z=36
  - intersection point is $x_1=2, x_2=6$



Linear Programs - example 2

- objective
  $z=3x_1+5x_2$
  - 4 objective lines drawn: z=0,15,25,36
- last z line intersecting feasible reagion: z=36
  - intersection point is $x_1=2, x_2=6$



Linear Programs - solution

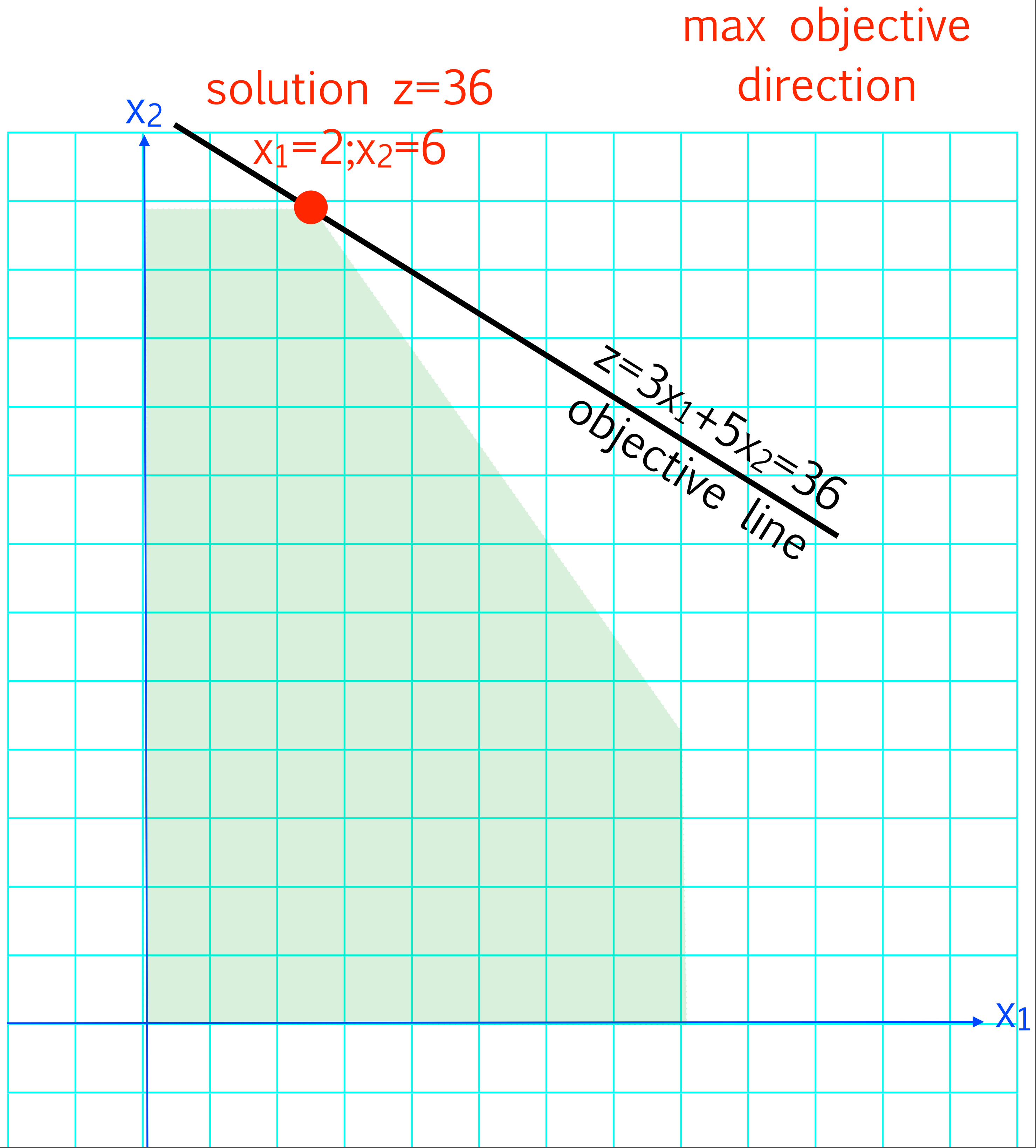
maximize

$$z = 3x_1 + 5x_2$$

subject to

$$x_1 \le 4$$
$$2x_2 \le 12$$
$$3x_1 + 2x_2 \le 18$$
$$x_1, x_2 \ge 0$$

- last y line intersecting feasible reagion: z=36
  - intersection point is $x_1=2, x_2=6$



Linear Programs - solution

maximize

$$z = 3x_1 + 5x_2$$

subject to

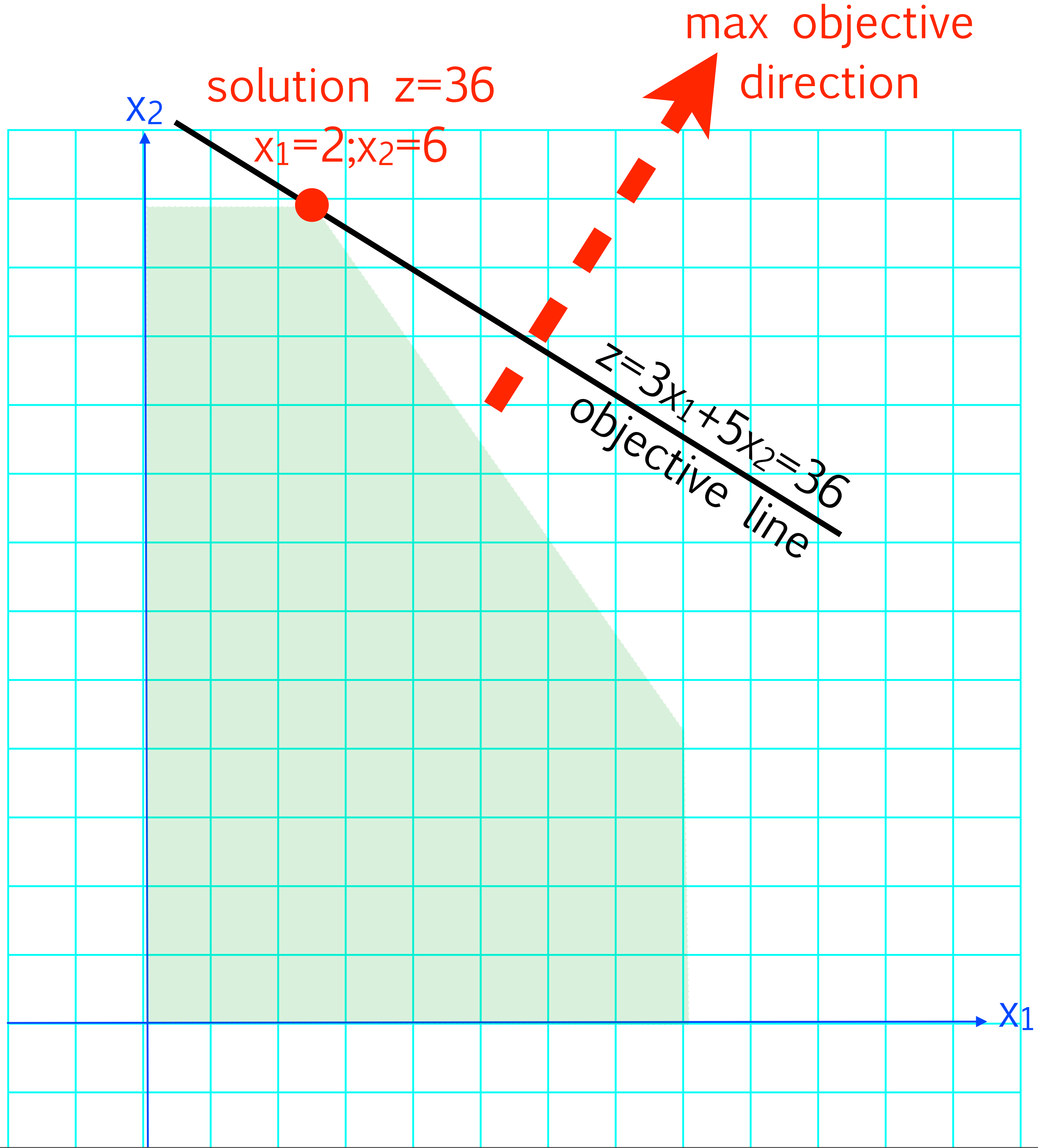
$$x_1 \leq 4$$

$$2x_2 \leq 12$$

$$3x_1 + 2x_2 \leq 18$$

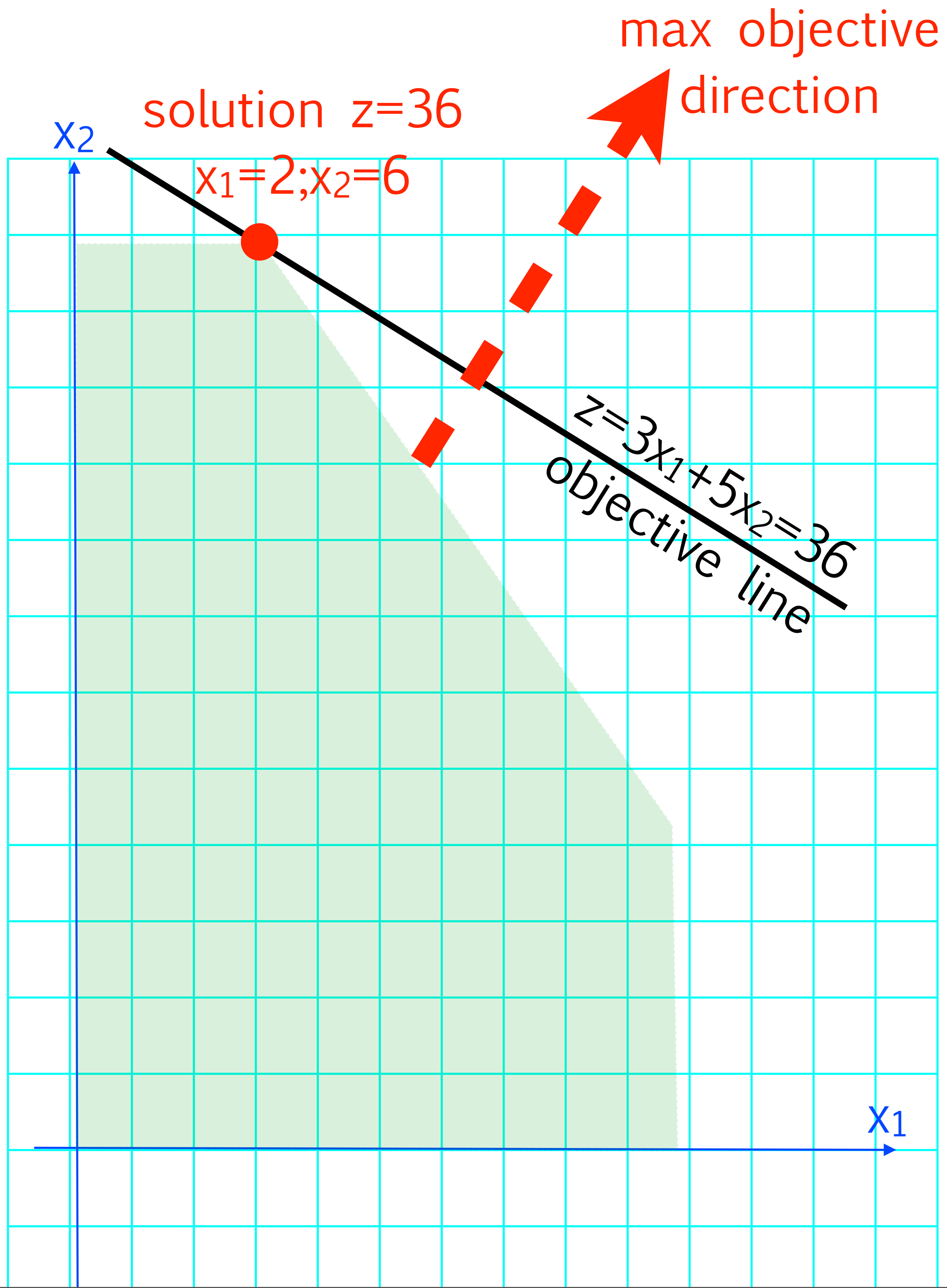
$$x_1, x_2 \geq 0$$

- last y line intersecting feasible reagion: z=36
  - intersection point is $x_1=2, x_2=6$



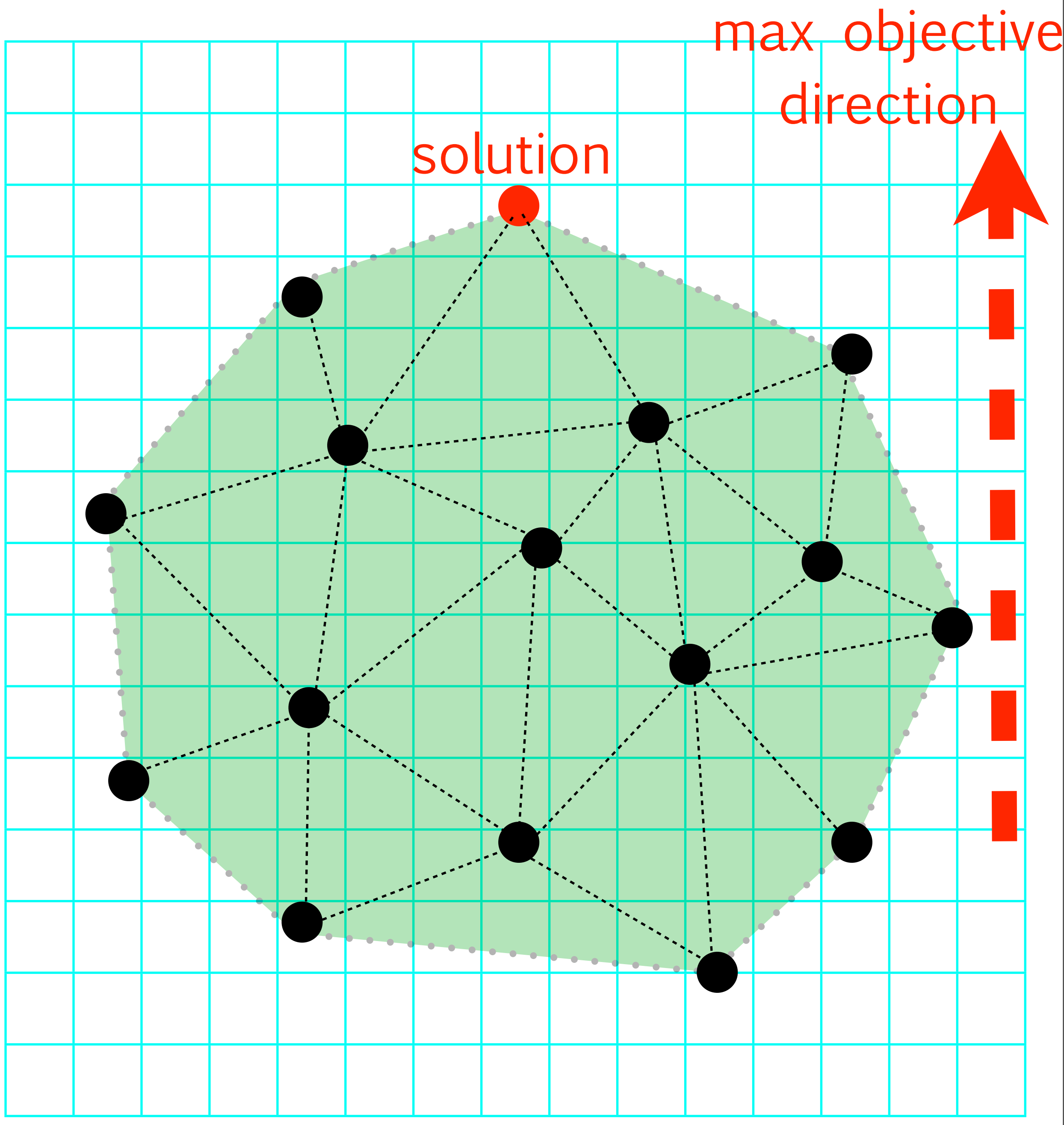
LP - solution critical observations

- OBSERVATION 1: the solution is in a corner(vertex) of the feasible region
- precisely the corner that is furtest in the direction of max objective



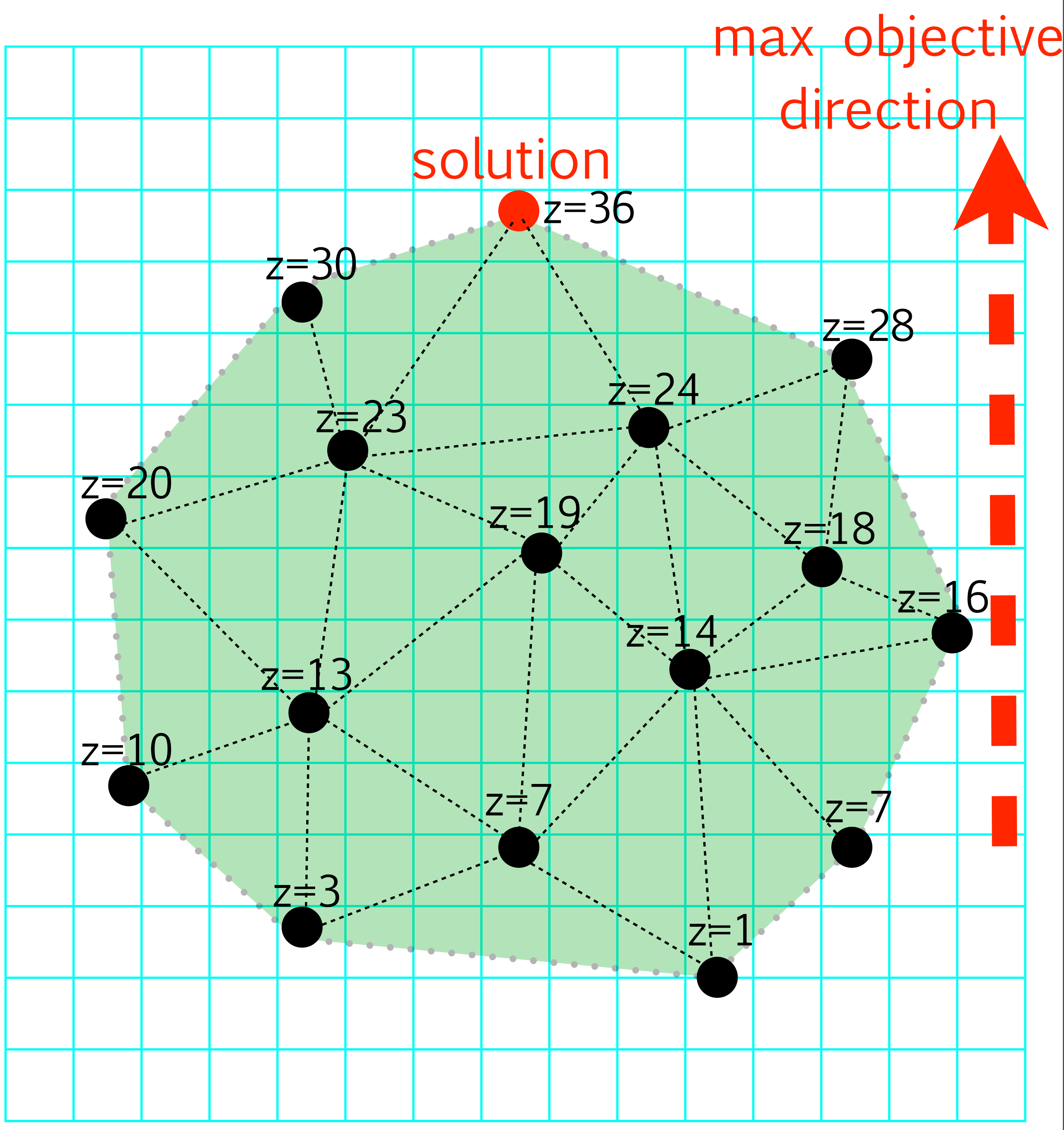
LP - solution critical observations

- OBSERVATION 2: feasible region is a convex polygon multidimensional
  - think of a ball in 3 dimensions, only not round but with triangle sides



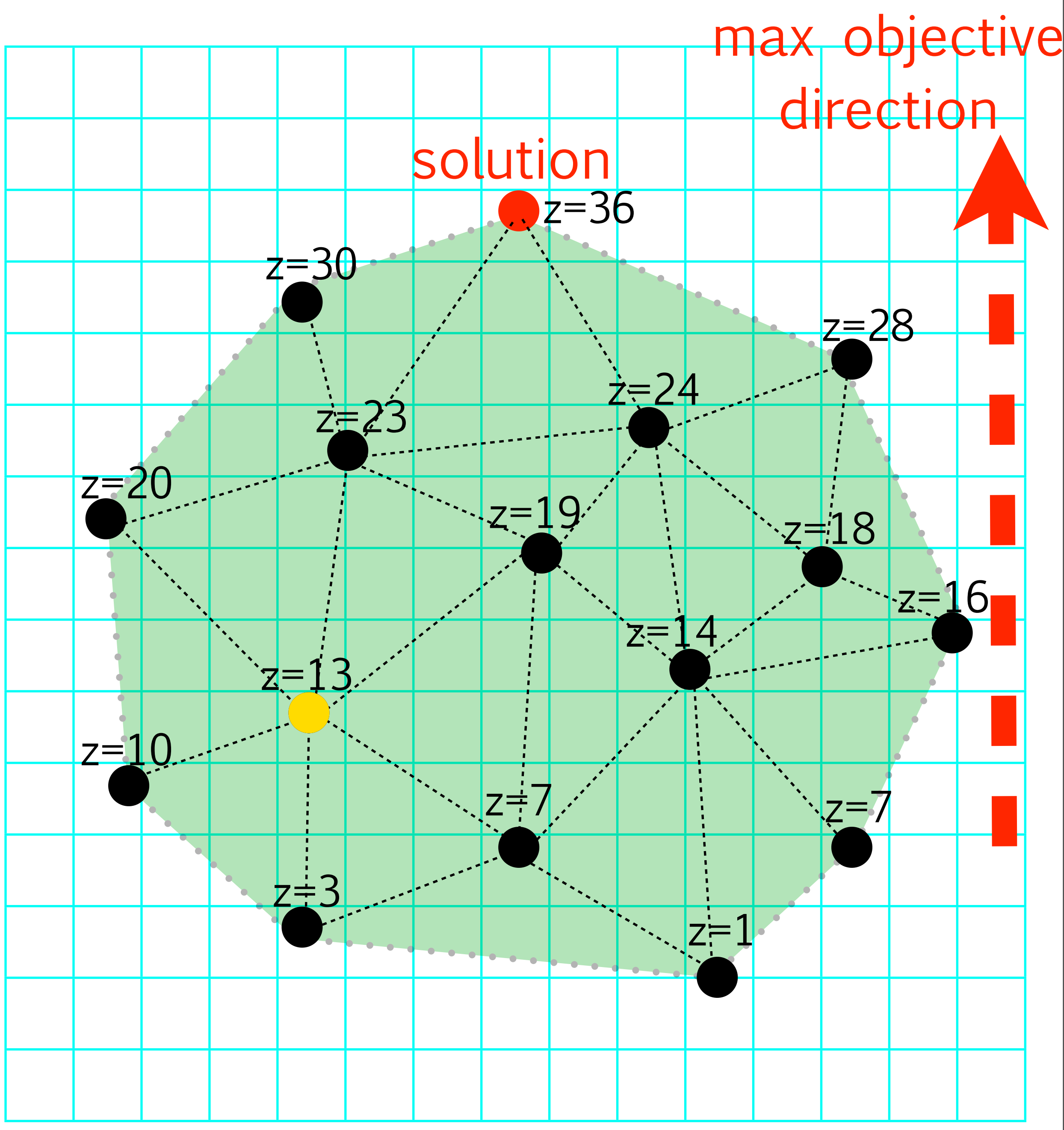
LP - solution critical observations

- OBSERVATION 2: feasible region is a convex polygon multidimensional
  - think of a ball in 3 dimensions, only not round but with triangle sides
- write objective for each corner



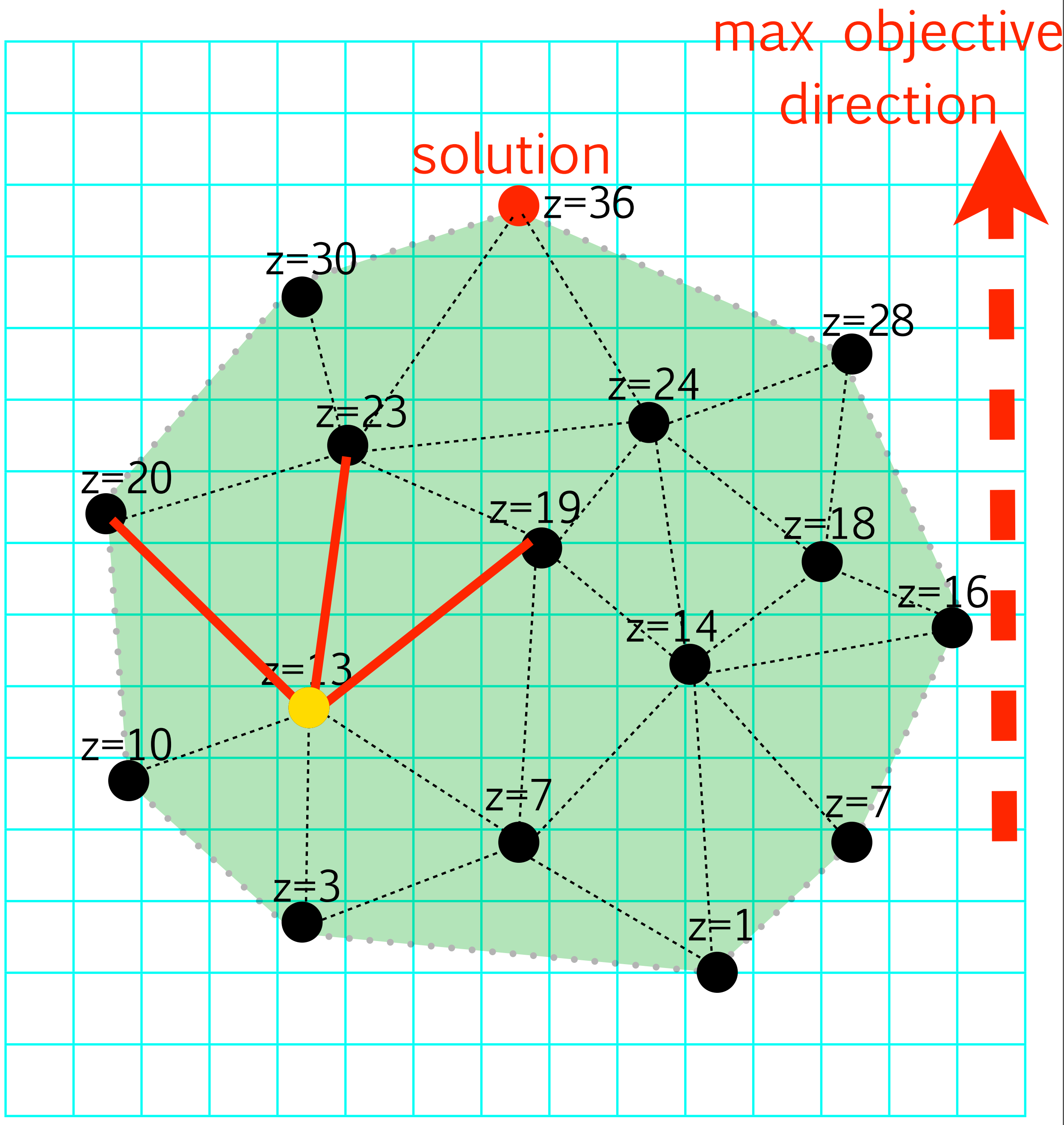
LP - solution critical observations

- OBSERVATION 2: feasible region is a convex polygon multidimensional
  - think of a ball in 3 dimensions, only not round but with triangle sides
- write objective for each corner
- convexity means that each vertex has :



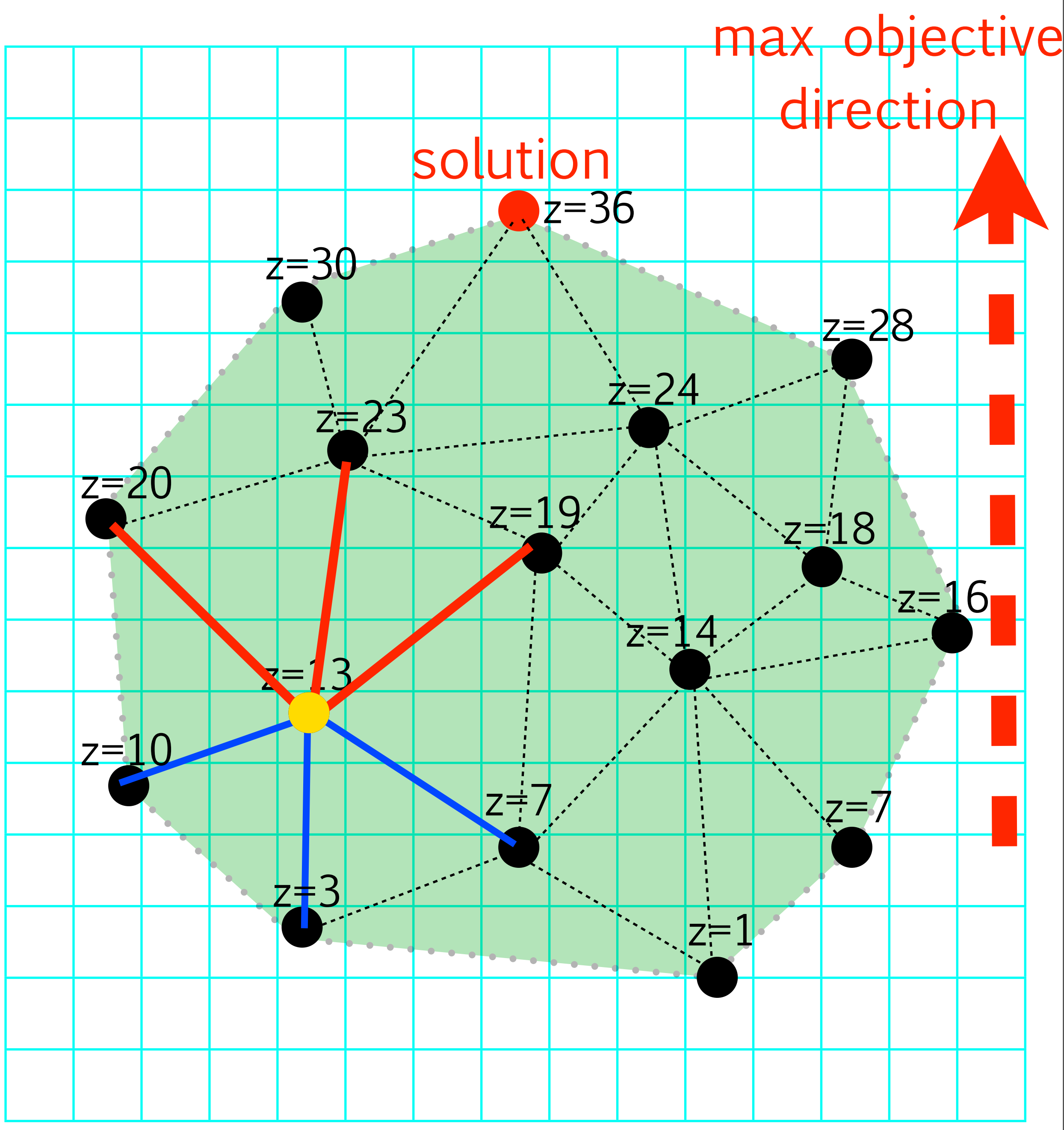
LP - solution critical observations

- OBSERVATION 2: feasible region is a convex polygon multidimensional
  - think of a ball in 3 dimensions, only not round but with triangle sides
- write objective for each corner
- convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)



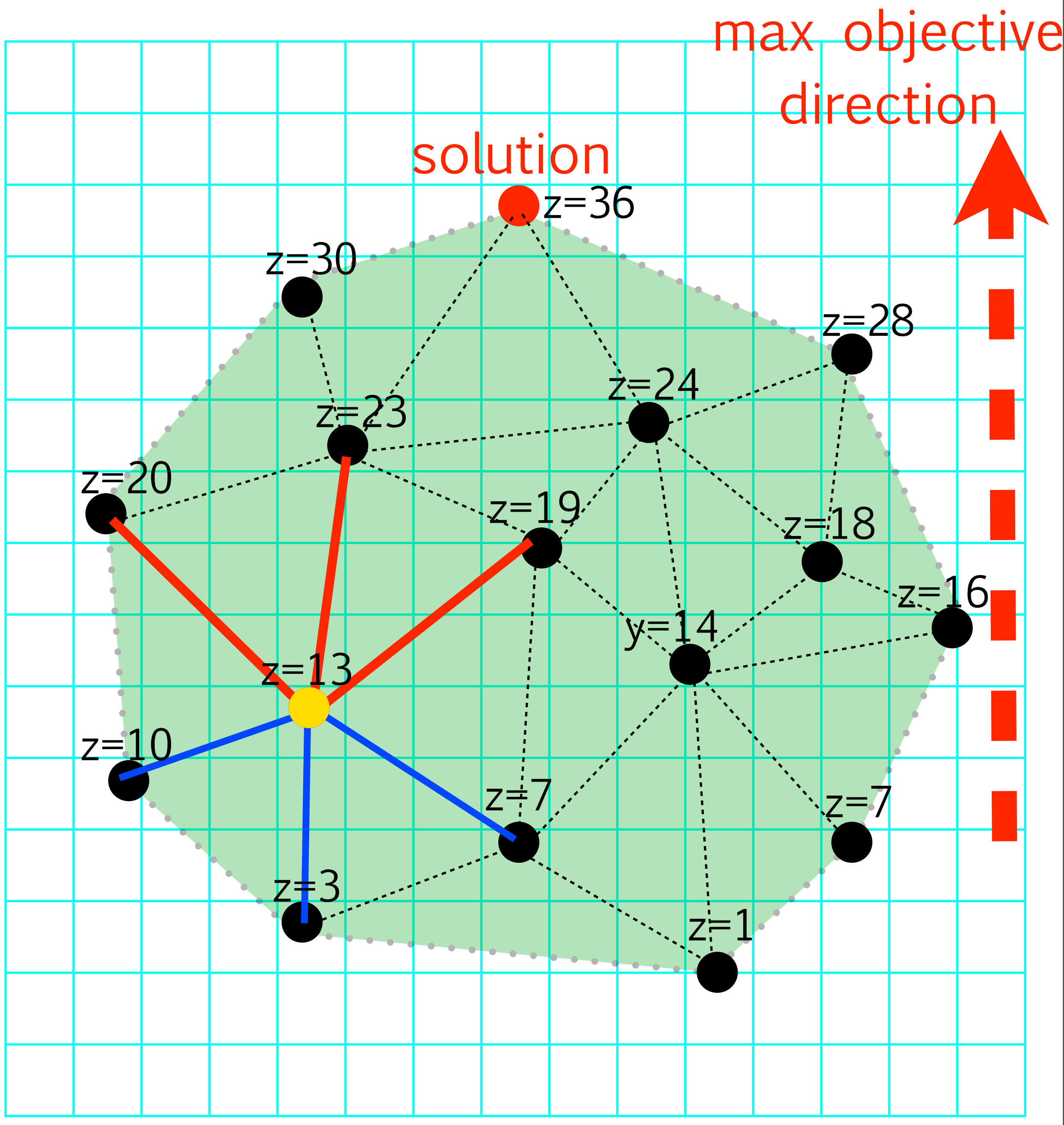
LP - solution critical observations

- OBSERVATION 2: feasible region is a convex polygon multidimensional
  - think of a ball in 3 dimensions, only not round but with triangle sides
- write objective for each corner
- convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)



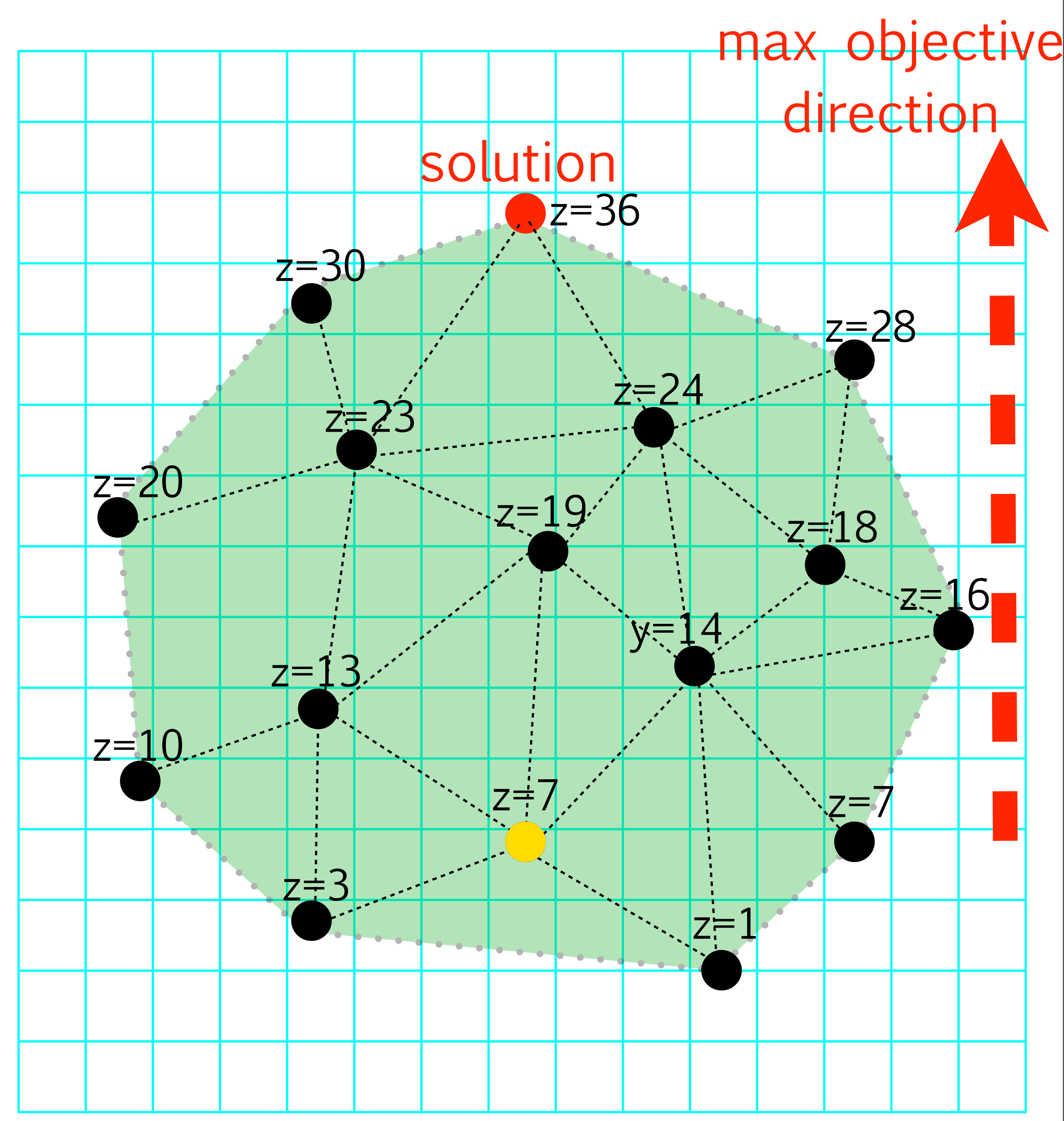
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)



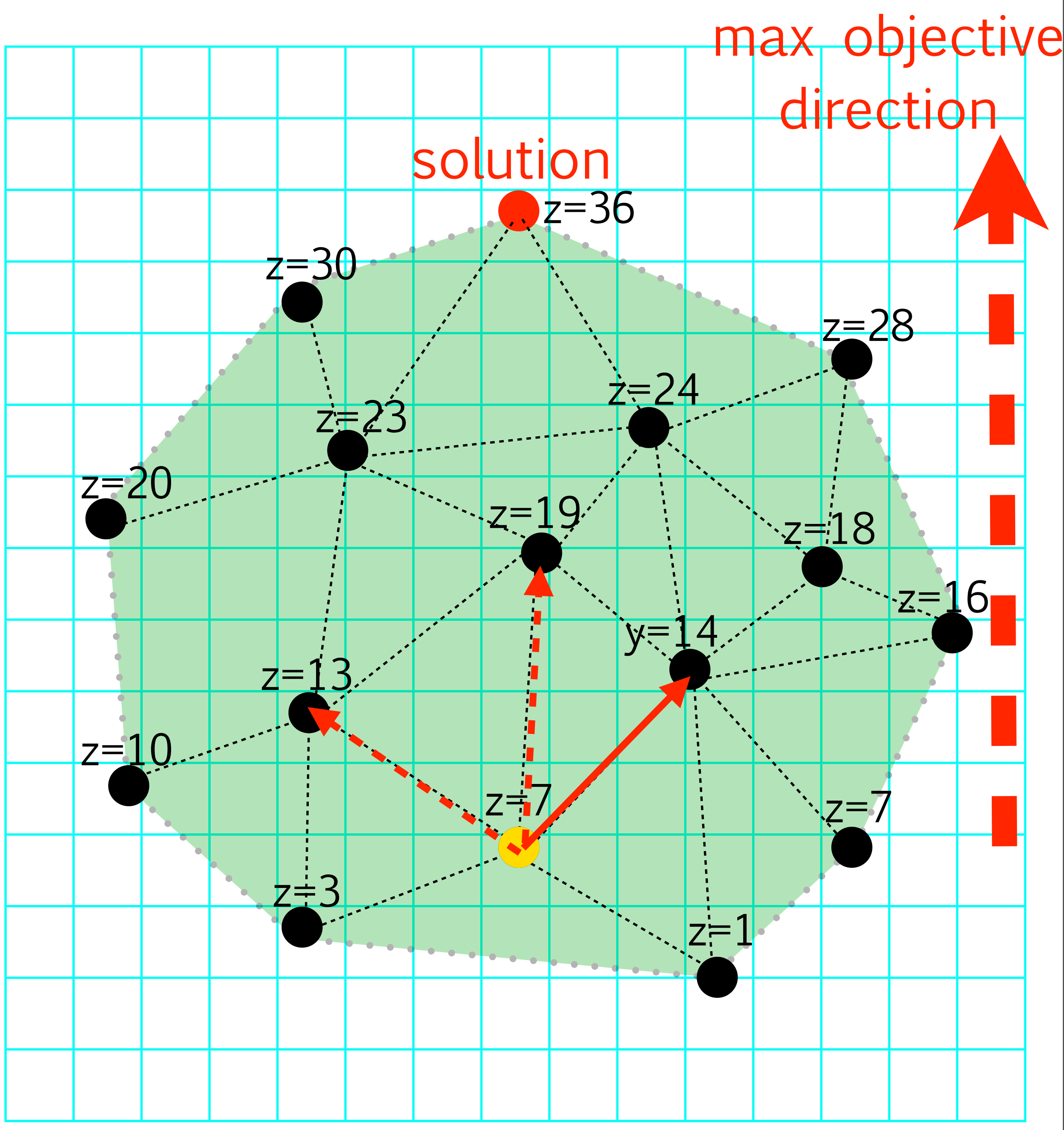
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR



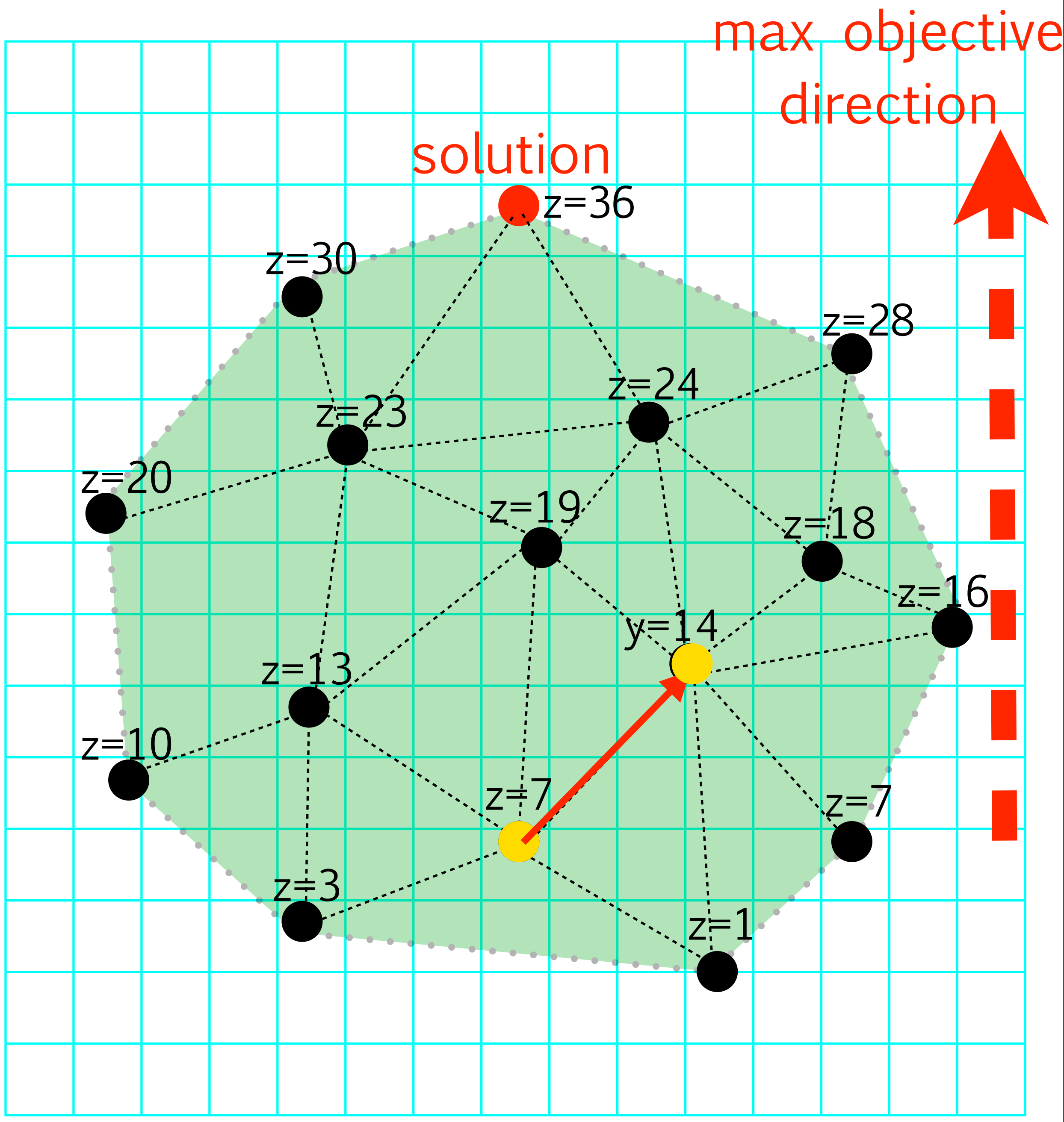
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR
- "walk" to any adjacent corner with higher objective



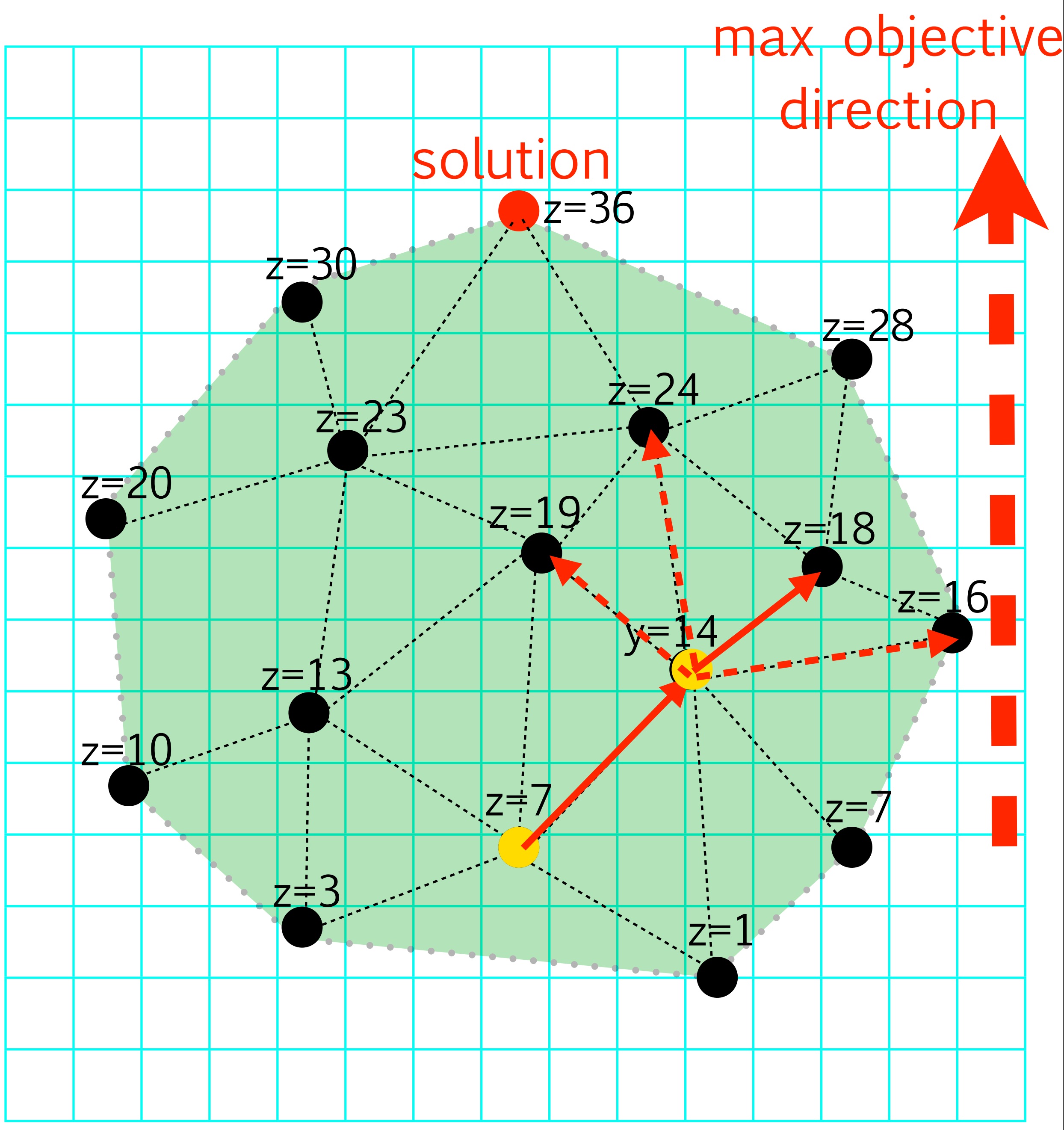
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR
- "walk" to any adjacent corner with higher objective



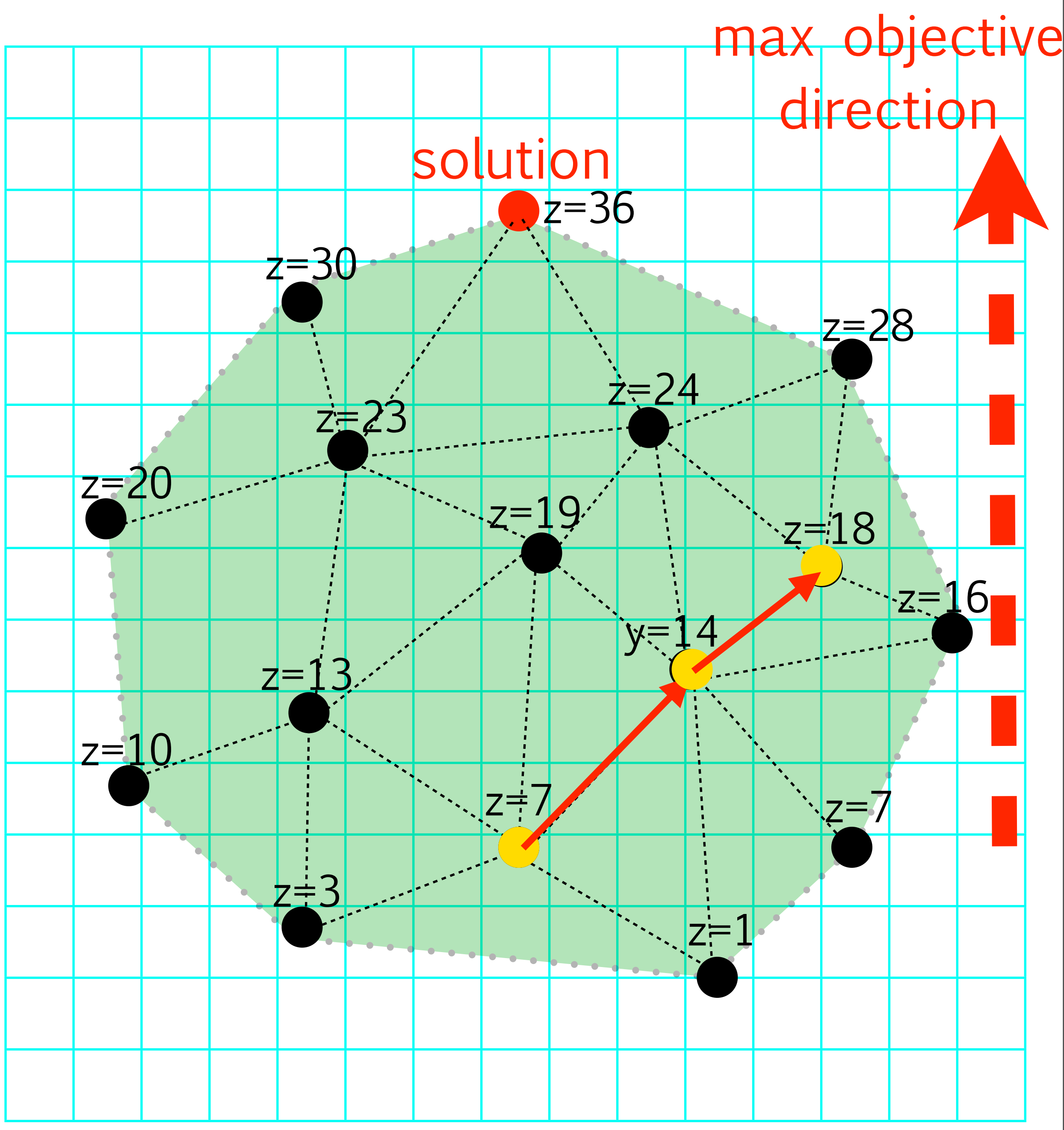
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR
- "walk" to any adjacent corner with higher objective
- repeat



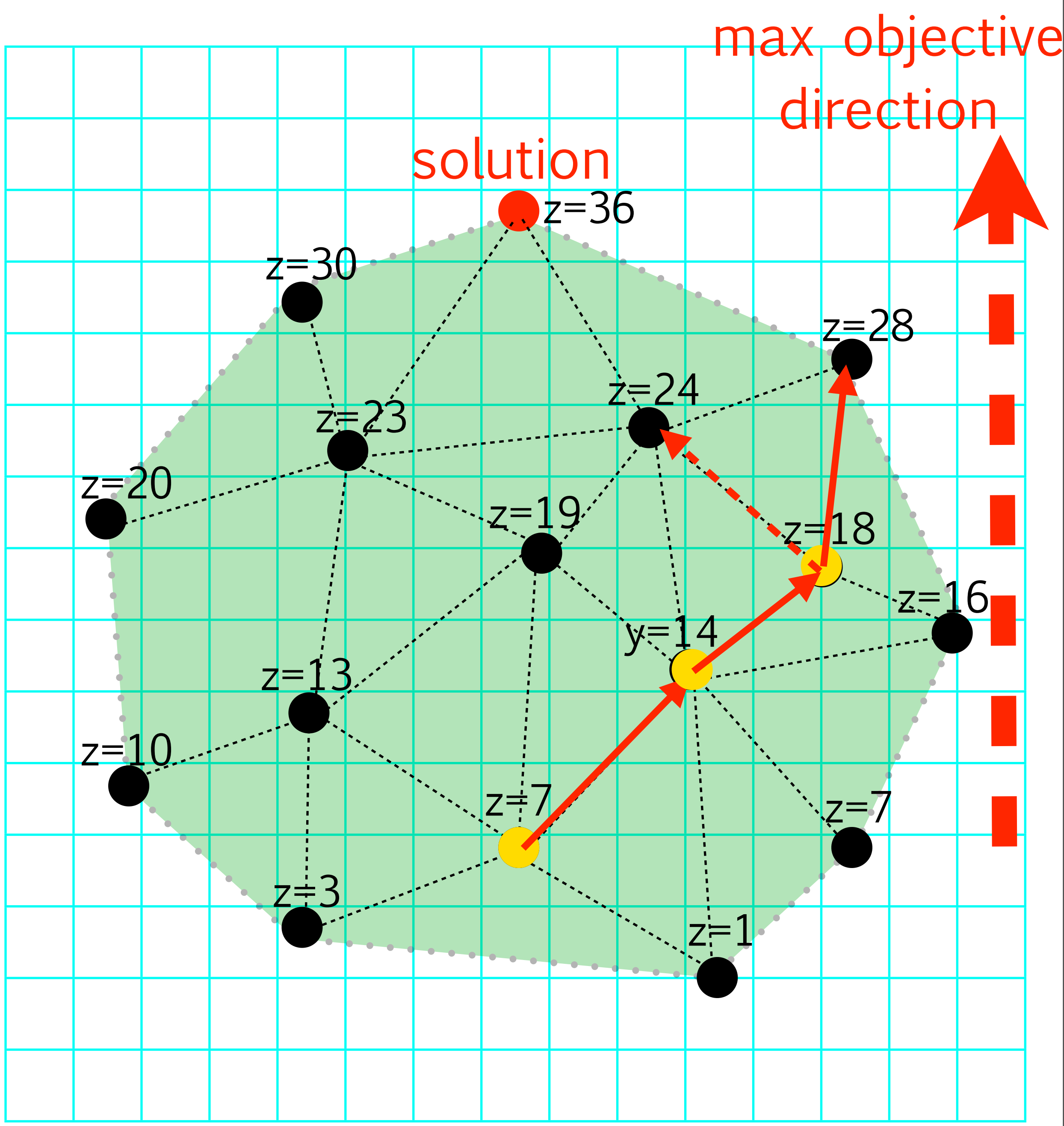
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR
- "walk" to any adjacent corner with higher objective
- repeat



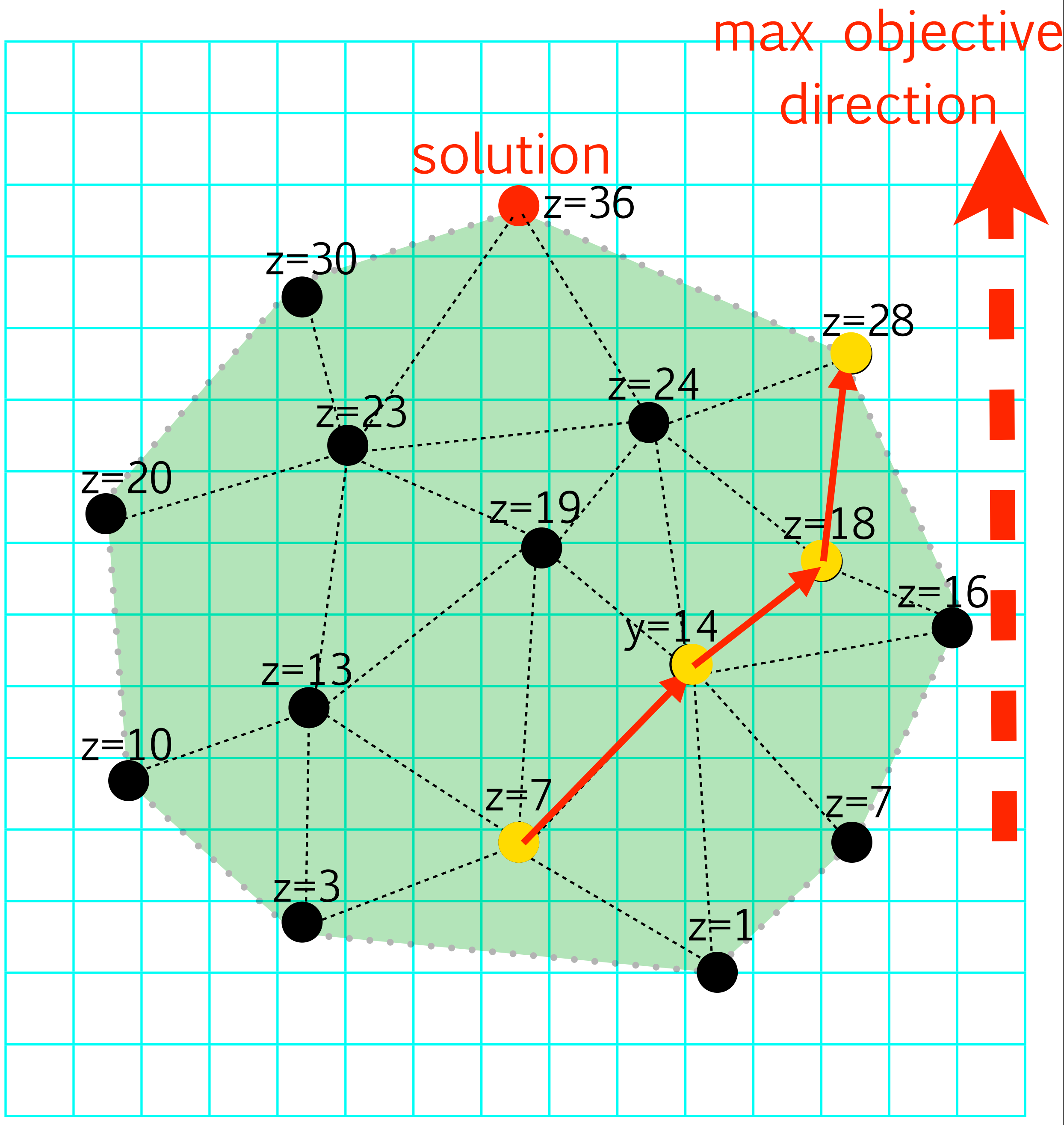
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR
- "walk" to any adjacent corner with higher objective
- repeat



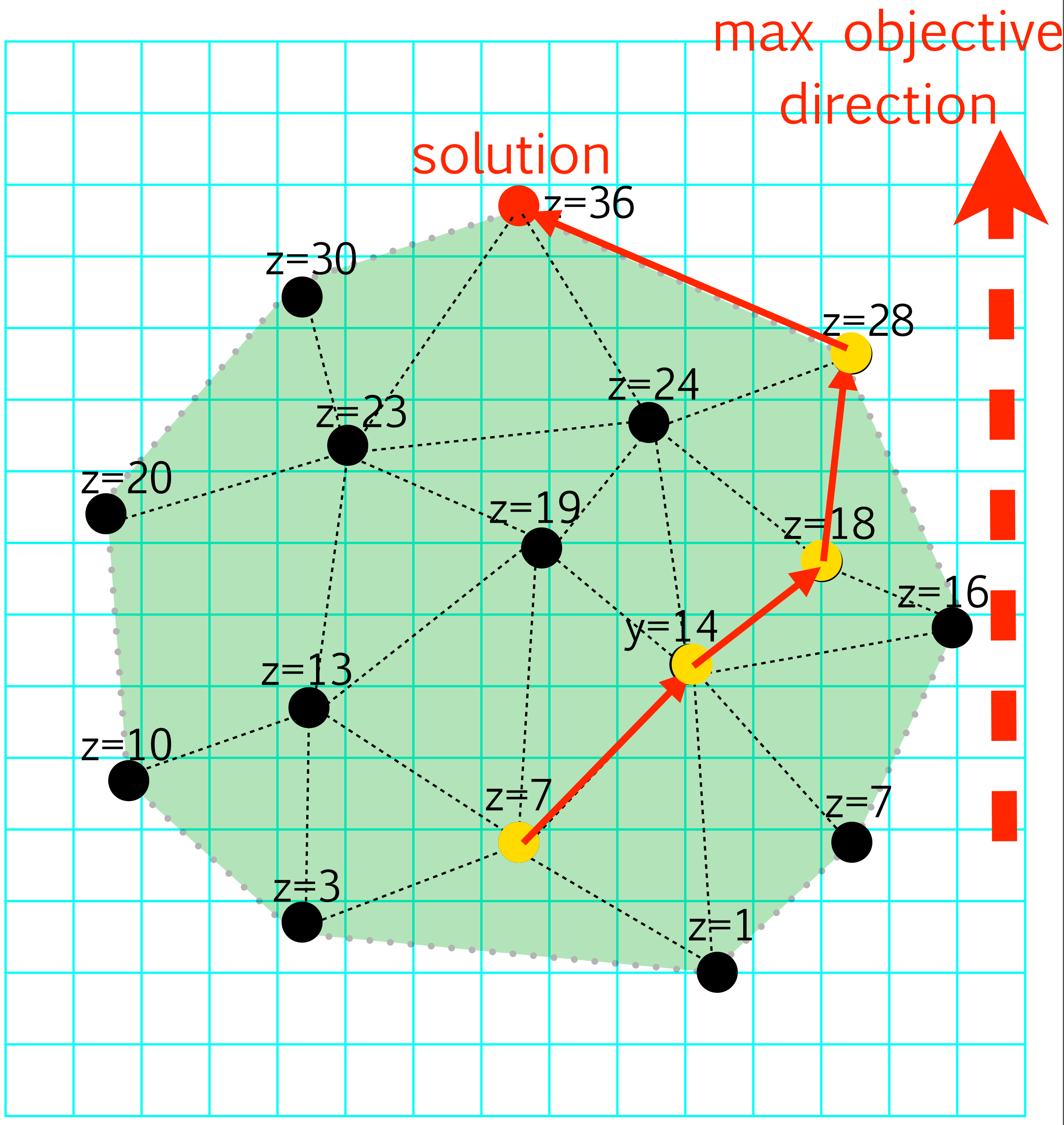
LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR
- "walk" to any adjacent corner with higher objective
- repeat



LP - simplex algorithm idea

- feasible region (FR) convexity means that each vertex has :
  - higher obj neighbors in the max-obj direction (red)
  - lower obj neighbors in opposite direction (blue)
- idea: start in any corner of FR
- "walk" to any adjacent corner with higher objective
- repeat
- stop when there is no higher-obj neighbor: we found the solution



LP examples: Shortest Path as LP

$$\begin{aligned} &\text{maximize} \quad d_t \\ &\text{subject to} \\ &\qquad d_v \le d_u + w(u,v) \quad \text{for each edge } (u,v) \in E \\ &\qquad d_s = 0\,. \end{aligned}$$

- Graph G=(V,E) with weighted edges given by w
- s=source; t= sink
- distance d_t from s to t is maximized (objective) but each d_v restricted to not more than d_u + edge-w(u,v)
- exercise: explain why this linear program finds the shortest path from s to t

LP examples: Maximum Flow as LP

$$\text{maximize} \quad \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs}$$

subject to

$$\begin{aligned}
f_{uv} &\leq c(u,v) && \text{for each } u, v \in V , \\
\sum_{v \in V} f_{vu} &= \sum_{v \in V} f_{uv} && \text{for each } u \in V - \{s,t\} \\
f_{uv} &\geq 0 && \text{for each } u, v \in V .
\end{aligned}$$

- Graph G(V,E), c(u,v) = capacity of edge (u,v)
- s= source, t=sink
- f_{uv} is the flow on edge u,v
- constraints are given by symmetry, and edge capacities
- objective is the flow from the source

Standard Form

$$
\begin{array}{lrcrcrcr}
\text{maximize} & 2x_1 & - & 3x_2 & + & 3x_3 & & \\
\text{subject to} & & & & & & & \\
& x_1 & + & x_2 & - & x_3 & \leq & 7 \\
& -x_1 & - & x_2 & + & x_3 & \leq & -7 \\
& x_1 & - & 2x_2 & + & 2x_3 & \leq & 4 \\
& & x_1, x_2, x_3 & & & & \geq & 0
\end{array}
$$

- objective is always "maximize" (not "minimize")
- all variables are constrained to be positive
- all constraints (other than positive variables) are "≤", none is "≥"
- book discusses simple steps/arithmetic to get any linear problem into standard form

Standard Form

- book discusses simple steps/arithmetic to get any linear problem into standard form
  - if objective is "minimize", reverse the objective sign
  - if a constraint is "equal to", replace it with 2 constraints "$\leq$" and "$\geq$"
  - if a constraint is "$\geq$", reverse the signs to make it "$\leq$"
  - if a variable does not have the nonnegativity constraint, replace it with a difference of two new variables, and add constraints that these two variables are nonnegative.

Slack Form

$$
\begin{array}{lrcrcrcrcr}
\text{maximize} & & & & & 2x_1 & - & 3x_2 & + & 3x_3 \\
\text{subject to} & & & & & & & & & \\
& x_4 & = & 7 & - & x_1 & - & x_2 & + & x_3 \\
& x_5 & = & -7 & + & x_1 & + & x_2 & - & x_3 \\
& x_6 & = & 4 & - & x_1 & + & 2x_2 & - & 2x_3 \\
& & & x_1, x_2, x_3, x_4, x_5, x_6 & & & \geq & 0 & . &
\end{array}
$$

- same as standard form, plus...
- ... all constraints (other than x≥0) are equalities
  - book discusses the easy steps to get the system in slack form
- basic variables : right side of constraints, typically present in objective
- nonbasic variables: left side of constraints, not part of the objective

Slack Form with matrices

$$
\begin{aligned}
z &= 28 - \frac{x_3}{6} - \frac{x_5}{6} - \frac{2x_6}{3} \\
x_1 &= 8 + \frac{x_3}{6} + \frac{x_5}{6} - \frac{x_6}{3} \\
x_2 &= 4 - \frac{8x_3}{3} - \frac{2x_5}{3} + \frac{x_6}{3} \\
x_4 &= 18 - \frac{x_3}{2} + \frac{x_5}{2} ,
\end{aligned}
$$

- $x \geqslant 0$ implicit, no need to write it
- z is the objective to be maximized
- no need for "subject to", just list the constraints
- B = basic variables set = {3,5,6}
- N = nonbasic variables set = {4,2,4}
- constraints in matrix form $Ax \leq b$
  - A= constraints coefficients (matrix); b= constraints value (array)
- objective in matrix form cx
  - c = objective coefficients (array); v= free constant in objective

Simplex Algorithm

- N = { nonbasic variables indices};
- B = { basic variables indices};
  - $N \cup B = \{1, 2, \ldots, n+m\}$
- A = constraints coefficients
- c = objective coefficients
- b = constraints value
- v = constant term in the objective (if any)

Simplex Algorithm

$$\begin{array}{rcrcrcrcr} z & = & & & 3x_1 & + & x_2 & + & 2x_3 \\ x_4 & = & 30 & - & x_1 & - & x_2 & - & 3x_3 \\ x_5 & = & 24 & - & 2x_1 & - & 2x_2 & - & 5x_3 \\ x_6 & = & 36 & - & 4x_1 & - & x_2 & - & 2x_3 \end{array}$$

- start with a basic feasible solution, for example X=0;

Simplex Algorithm

$$
\begin{aligned}
z &= && 3x_1 &&+ x_2 &&+ 2x_3 \\
x_4 &= 30 &&- x_1 &&- x_2 &&- 3x_3 \\
x_5 &= 24 &&- 2x_1 &&- 2x_2 &&- 5x_3 \\
x_6 &= 36 &&- 4x_1 &&- x_2 &&- 2x_3
\end{aligned}
$$

- start with a basic feasible solution, for example X=0;
- pick a basic variable with positive coefficient in objective, say x1
  - increase that basic var until one of the nonbasic x becomes 0
  - in our example X6 becomes 0 first, when x1=9; x6 equation called "tight"

Simplex Algorithm

$$
\begin{aligned}
z &= && 3x_1 + x_2 + 2x_3 \\
x_4 &= 30 &-& x_1 - x_2 - 3x_3 \\
x_5 &= 24 &-& 2x_1 - 2x_2 - 5x_3 \\
x_6 &= 36 &-& 4x_1 - x_2 - 2x_3
\end{aligned}
$$

- start with a basic feasible solution, for example X=0;
- pick a basic variable with positive coefficient in objective, say x1
  - increase that basic var until one of the nonbasic x becomes 0
  - in our example X6 becomes 0 first, when x1=9; x6 equation called "tight"
- exchange/pivot x1 and x6
  - rewrite x1 from x6 tight equation

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

Simplex Algorithm

$$
\begin{aligned}
z &= && 3x_1 &+& x_2 &+& 2x_3 \\
x_4 &= 30 &-& x_1 &-& x_2 &-& 3x_3 \\
x_5 &= 24 &-& 2x_1 &-& 2x_2 &-& 5x_3 \\
x_6 &= 36 &-& 4x_1 &-& x_2 &-& 2x_3
\end{aligned}
$$

- start with a basic feasible solution, for example X=0;
- pick a basic variable with positive coefficient in objective, say x1
  - increase that basic var until one of the nonbasic x becomes 0
  - in our example X6 becomes 0 first, when x1=9; x6 equation called "tight"
- exchange/pivot x1 and x6
  - rewrite x1 from x6 tight equation

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

- recompute nonbasic var x4, x5 and the objective z using the x1 new formula
  - update N,B,A,C,b,v : new basic/nonbasic variables, different coefficients, etc

Simplex Algorithm

$$\begin{aligned} z &= 3x_1 + x_2 + 2x_3 \\ x_4 &= 30 - x_1 - x_2 - 3x_3 \\ x_5 &= 24 - 2x_1 - 2x_2 - 5x_3 \\ x_6 &= 36 - 4x_1 - x_2 - 2x_3 \end{aligned}$$

$$\begin{aligned} z &= 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_6}{4} \\ x_1 &= 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4} \\ x_4 &= 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_6}{4} \\ x_5 &= 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_6}{2} \end{aligned}$$

- start with a basic feasible solution, for example X=0;
- pick a basic variable with positive coefficient in objective, say x1
  - increase that basic var until one of the nonbasic x becomes 0
  - in our example X6 becomes 0 first, when x1=9; x6 equation called "tight"
- exchange/pivot x1 and x6
  - rewrite x1 from x6 tight equation

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

- recompute nonbasic var x4, x5 and the objective z using the x1 new formula
  - update N,B,A,C,b,v : new basic/nonbasic variables, different coefficients, etc

Simplex Algorithm

$$
\begin{aligned}
z &= 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_6}{4} \\
x_1 &= 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4} \\
x_4 &= 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_6}{4} \\
x_5 &= 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_6}{2}
\end{aligned}
$$

- repeat: pick a basic variable with positive coeficient in objective, say x3
  - increase that basic var until one of the nonbasic x becomes 0: X5 becomes 0 first; x5 equation is "tight"
- exchange/pivot x3 and x5
  - rewrite x3 from x5 tight equation $x_3 = 3/2 - 3x_2/8 - x_5/4 + x_6/8$.
  - recompute nonbasic var x1, x4 and the objective z using the x3 new formula
  - update N,B,A,C,b,v : new basic/nonbasic variables, different coefficients, etc

Simplex Algorithm

$$\begin{aligned} z &= 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_6}{4} \\ x_1 &= 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4} \\ x_4 &= 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_6}{4} \\ x_5 &= 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_6}{2} \end{aligned}$$

$$\begin{aligned} z &= \frac{111}{4} + \frac{x_2}{16} - \frac{x_5}{8} - \frac{11x_6}{16} \\ x_1 &= \frac{33}{4} - \frac{x_2}{16} + \frac{x_5}{8} - \frac{5x_6}{16} \\ x_3 &= \frac{3}{2} - \frac{3x_2}{8} - \frac{x_5}{4} + \frac{x_6}{8} \\ x_4 &= \frac{69}{4} + \frac{3x_2}{16} + \frac{5x_5}{8} - \frac{x_6}{16} \end{aligned}$$

- repeat: pick a basic variable with positive coeficient in objective, say x3
  - increase that basic var until one of the nonbasic x becomes 0: X5 becomes 0 first; x5 equation is "tight"
- exchange/pivot x3 and x5
  - rewrite x3 from x5 tight equation $x_3 = 3/2 - 3x_2/8 - x_5/4 + x_6/8$.
  - recompute nonbasic var x1, x4 and the objective z using the x3 new formula
  - update N,B,A,C,b,v : new basic/nonbasic variables, different coefficients, etc

Simplex Termination

- four possibilities:
- 1) didnt start (a feasible initial solution was not given)
  - return "infeasible"
- 2) at some iteration, all basic variable have negative coefficients
  - STOP: solution is obtained by setting the basic vars to 0, and compute the original variables
- 3) at some iteration, no constraint $x \geq 0$ is violated by increasing a basic var
  - STOP: the system is unbounded (objective can be increased to ∞)
- 4) Cycling back and forth between variable-values with no progress on objective
  - fix the algorithm, so this never happens

Simplex termination: cycling

- its possible that SIMPLEX starts cycling between some variables, without making progress
  - this can occur when multiple solutions realizes the maximum objective
- how to avoid this behavior: Bland's rule
  - when choosing variables, if ties exist, choose variables with the smallest index
  - thats when choosing basic var to increase
  - or when constraints become tight

SIMPLEX running time

- SIMPLEX terminates after at most $\binom{n+m}{m}$ iterations
  - Assuming a feasible initial solution
  - using Bland's rule to break ties
- exponential running time (worst case), but quite efficient in practice.
- under certain probabilistic assumptions of the input, SIMPLEX runs in expected polynomial time.
- variants of SIMPLEX on GRAPH/NEWORK problems run in polynomial time
  - shortest-paths, maximum-flow, minimum-cost-flow problems

Initial Feasible Solution

- initial feasible solution sometimes easy, set X=0
- sometimes tricky
- use a different "auxiliary" LP to determine if problem
  - is infeasible (no solution)
  - is feasible, obtain a slack form and initial feasible solution

Initial Feasible Solution

$$\begin{aligned}
\text{maximize} \quad & -x_0 \\
\text{subject to} \quad & \\
& \sum_{j=1}^{n} a_{ij}x_j - x_0 \leq b_i \quad \text{for } i = 1, 2, \ldots, m \,, \\
& x_j \geq 0 \quad \text{for } j = 0, 1, \ldots, n \,.
\end{aligned}$$

- Auxiliary LP: add variable x_0
  - constraints add $-x_0$ to original LP, $x_0>0$
  - objective is $-x_0$
- The original LP is feasible if and only if the auxiliary LP has the optimal solution with max objective $x_0=0$
  - optimal solution to aux LP with $x_0=0$ includes a feasible solution to original LP in x_1, x_2, x_3,...
  - the auxiliary LP has a feasible initial solution when x_0 small enough; from there it can be solved using SIMPLEX

Fundamental Theorem of LP

- Any linear program, either:
  - has an optimal solution with finite objective value. SIMPLEX returns such a solution (might be one of the many optimal solutions)
  - is infeasible, or no solution satisfies the constraints. SIMPLEX returns "infeasible"
  - is unbounded (objective can reach any high value). SIMPLEX returns "unbounded"