Branch & bound

[schema, value] = bb(F, sch, bnd)

- [v_{rx}, rsch] = relax(F, sch)
- if integer(rsch): return [rsch, v_{rx}]
- if $v_{rx} \geq$ bnd: return [sch, v_{rx}]
- Pick variable x_i
- [$sch^{(0)}$, $v^{(0)}$] = bb(F, sch/($x_i \rightarrow 0$), bnd)
- [$sch^{(1)}$, $v^{(1)}$] = bb(F, sch/($x_i \rightarrow 1$), min(bnd, $v^{(0)}$))
- if ($v^{(0)} \leq v^{(1)}$): return [$sch^{(0)}$, $v^{(0)}$]
- else: return [$sch^{(1)}$, $v^{(1)}$]

A random 3-CNF formula

$$
\begin{aligned}
&(x_5 \vee x_1 \vee x_2) \wedge (x_7 \vee x_2 \vee \bar{x}_4) \wedge (x_5 \vee x_2 \vee \bar{x}_8) \wedge (\bar{x}_6 \vee \bar{x}_1 \vee \bar{x}_7) \\
\wedge\, &(x_1 \vee x_3 \vee x_5) \wedge (\bar{x}_7 \vee x_1 \vee \bar{x}_6) \wedge (x_8 \vee x_5 \vee x_7) \wedge (\bar{x}_4 \vee \bar{x}_6 \vee \bar{x}_7) \\
\wedge\, &(\bar{x}_7 \vee x_2 \vee x_1) \wedge (\bar{x}_6 \vee x_4 \vee \bar{x}_4) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_2) \wedge (x_4 \vee x_2 \vee \bar{x}_1) \\
\wedge\, &(x_1 \vee \bar{x}_6 \vee x_6) \wedge (x_7 \vee \bar{x}_8 \vee \bar{x}_3) \wedge (x_3 \vee \bar{x}_4 \vee x_4) \wedge (\bar{x}_4 \vee x_7 \vee \bar{x}_3) \\
\wedge\, &(x_2 \vee x_4 \vee x_1) \wedge (\bar{x}_6 \vee \bar{x}_7 \vee x_5) \wedge (\bar{x}_2 \vee x_7 \vee \bar{x}_4) \wedge (\bar{x}_5 \vee x_6 \vee x_3) \\
\wedge\, &(x_7 \vee \bar{x}_1 \vee x_6) \wedge (x_7 \vee x_4 \vee x_7) \wedge (\bar{x}_5 \vee \bar{x}_6 \vee x_5) \wedge (x_7 \vee x_8 \vee \bar{x}_1) \\
\wedge\, &(\bar{x}_1 \vee \bar{x}_1 \vee x_3) \wedge (\bar{x}_8 \vee x_3 \vee \bar{x}_3) \wedge (x_5 \vee x_4 \vee \bar{x}_6) \wedge (x_4 \vee \bar{x}_1 \vee x_4) \\
\wedge\, &(\bar{x}_8 \vee x_4 \vee x_4) \wedge (\bar{x}_4 \vee \bar{x}_4 \vee \bar{x}_1) \wedge (\bar{x}_8 \vee x_7 \vee x_7) \wedge (\bar{x}_2 \vee x_8 \vee \bar{x}_8) \\
&\wedge (x_1 \vee x_2 \vee x_6) \wedge (\bar{x}_5 \vee \bar{x}_2 \vee x_1)
\end{aligned}
$$

Example search tree

Ordering rules

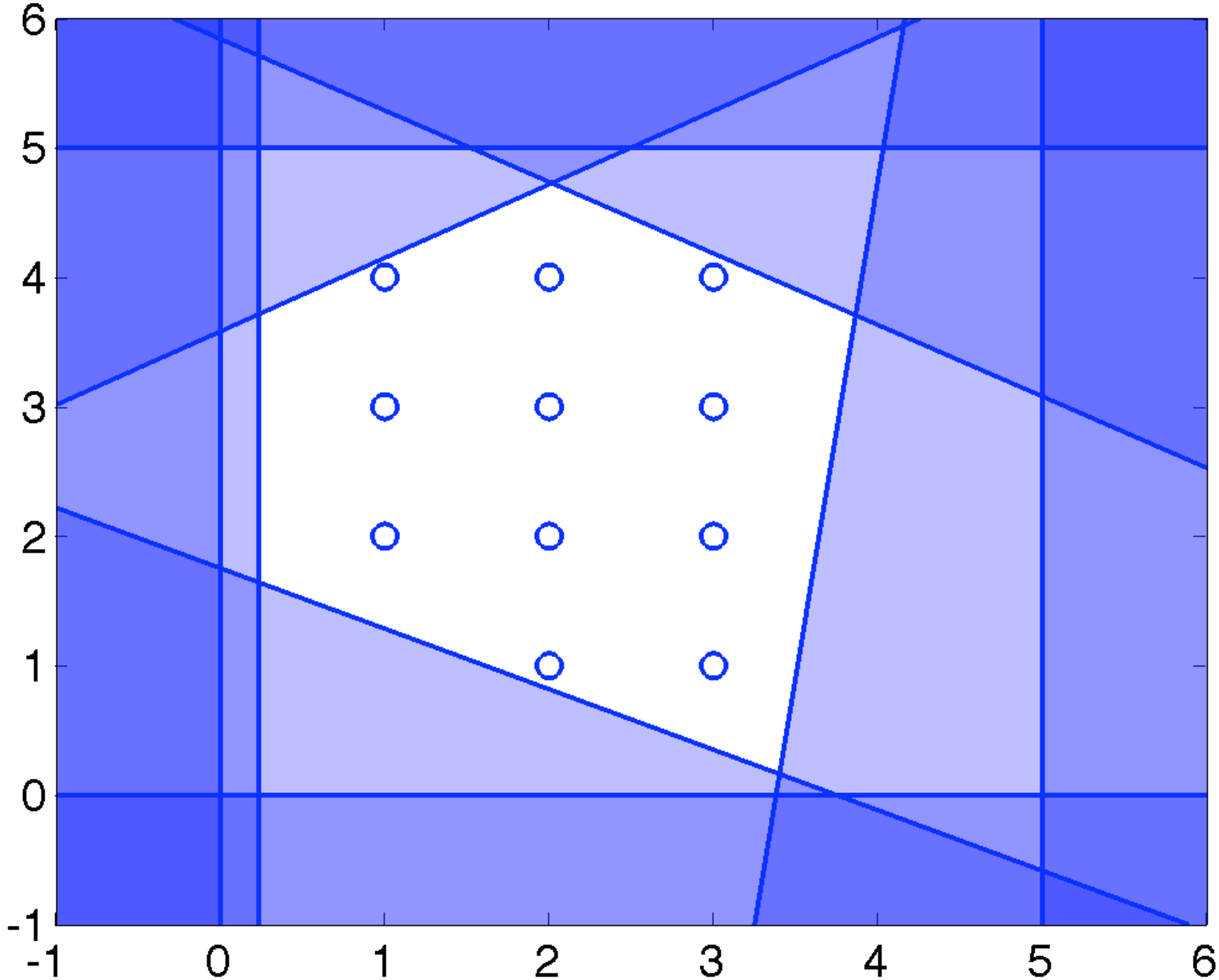
- If relaxation is available:
  - most certain variable first
  - most uncertain variable first
- If no relaxation:
  - most constrained variable first (fewest remaining values in domain)
  - activity rules (branch on variables that are “near” recent vars)

Summary so far

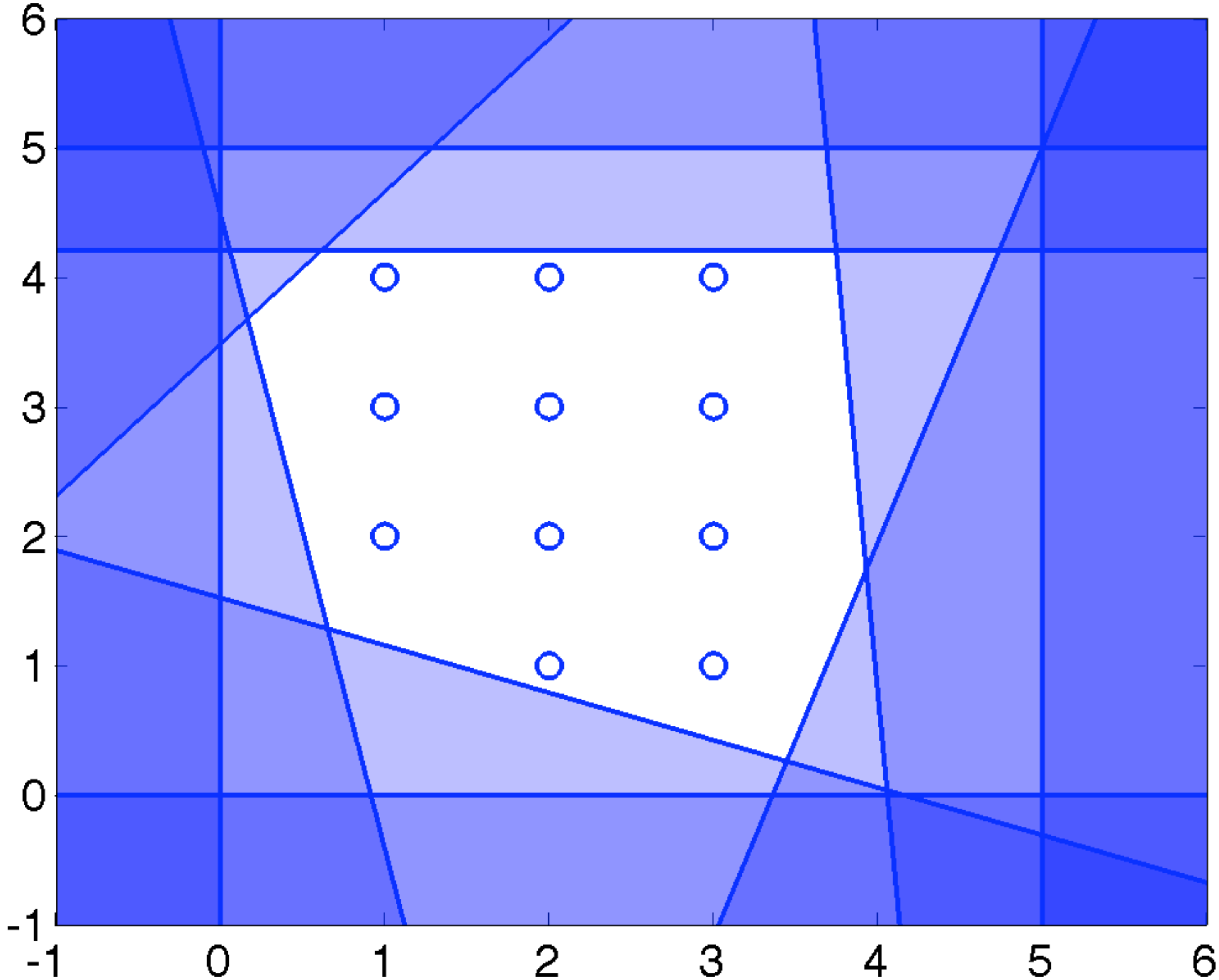
- Simple search
- Constraint propagation
- Branch & bound

Multiple representations

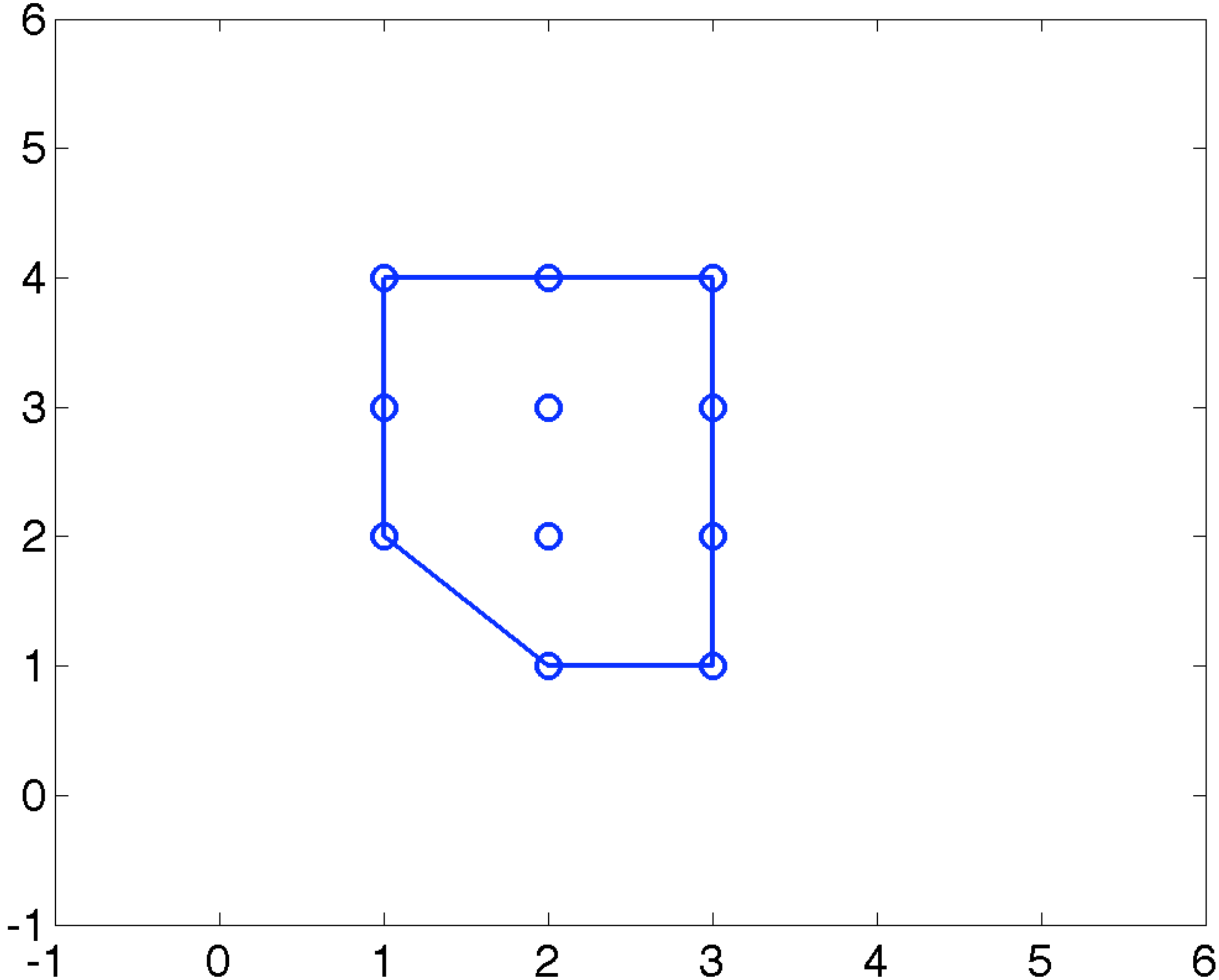
- Any given feasible region may have many different representations
- Can make problem much easier or harder to solve



6
5
4
3
2
1
0
-1
-1
0
1
2
3
4
5
6



6
5
4
3
2
1
0
-1
-1
0
1
2
3
4
5
6



6
5
4
3
2
1
0
-1
-1
0
1
2
3
4
5
6

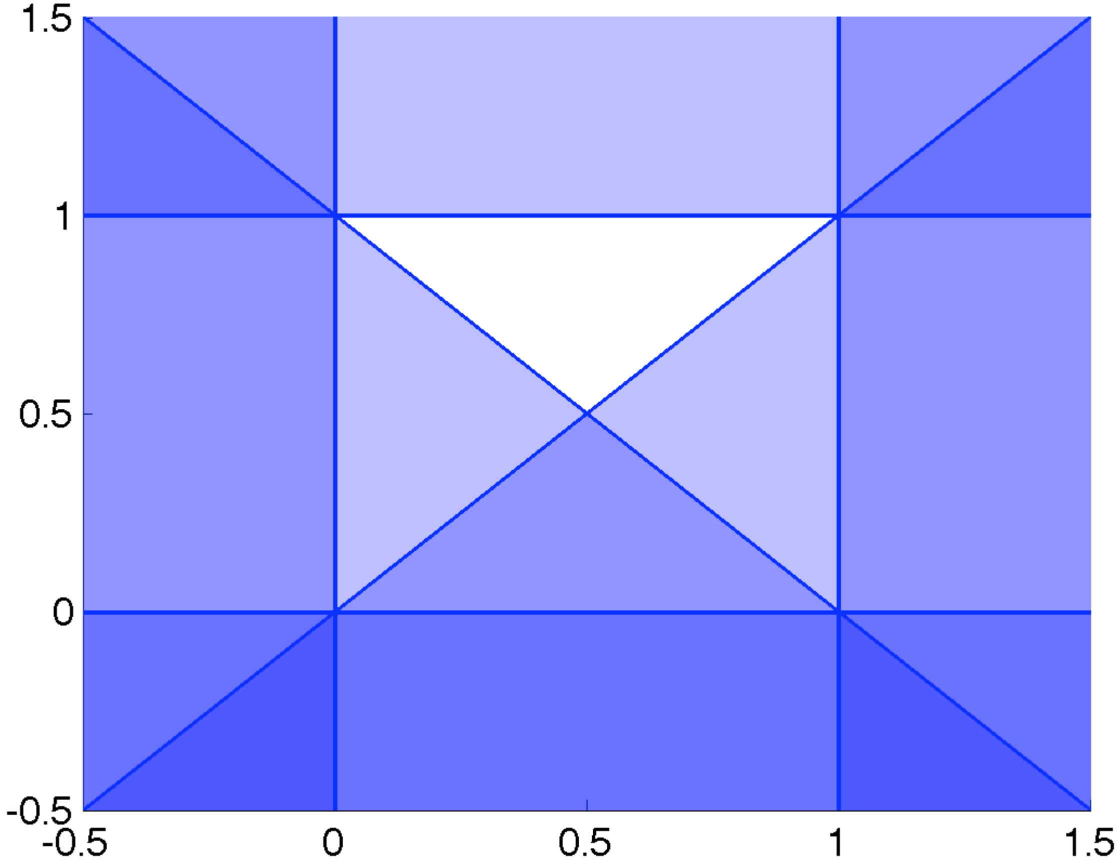
Multiple representations

- Typically, tension btwn tight & small
- Tightest: hull of integer feasible points
  - not small: can be exponentially many faces
- If we have the exact convex hull:

- So:
- Few variables, lots of constraints:

Cutting planes example

min y st $(1-x) + y \geq 1$, $x + y \geq 1$, $x,y \in \{0,1\}$



Resolution

$(a \vee \neg b \vee c) \wedge (\neg a \vee c \vee d)$

$\Rightarrow (\neg b \vee c \vee d)$

SAT and cutting planes

- These “resolution cuts” provide a partial description of the convex hull of integer feasible points for any SAT problem
- [Hooker 92]: can generalize to get a complete description
- Size:

Finding the convex hull

- If we have a non-integral optimal basic solution to current relaxation, we know that a cutting plane always exists
- But it might be difficult to find
- Interesting Q: is there a general way to find a cutting plane?

Summary so far

- Several improvements on simple search
  - constraint propagation
  - branch & bound
  - cutting planes
- B&B and cuts are very different
  - for a given problem, one can work much better than other
- Can we get best of both?

Branch & cut

[schema, value] = bc(F, sch, bnd)

- repeat until (no cuts added)
  - [v_{rx}, rsch] = relax(F, sch)
  - if integer(rsch): return [rsch, v_{rx}]
  - if $v_{rx} \geq$ bnd: return [sch, v_{rx}]
  - If desired: F := F ∪ {new cuts based on rsch}
- … continue as for branch & bound (try both branches, return better one)

Branch & cut discussion

- Don't always need to solve relaxation to find cuts
  - e.g., on failure in a SAT problem, know a subset of our decisions led to contradiction
- If we find a good cut near leaves of search tree, can sometimes "lift" it to apply to ancestor nodes