15-780: Grad AI
Lecture 7: Optimization, Games

Geoff Gordon (this lecture)
Ziv Bar-Joseph
TAs Geoff Hollinger, Henry Lin

Admin

- *Questions on HW1?*

Review

Linear and PO planners

- *Linear planners*
    - *forward and backward chaining*
- *Partial-order planning*
    - *action orderings, open preconditions, guard intervals, plan refinement*
- *Monkey & bananas example*

Plan graphs

- *Plan graphs for propositional planning*
- *How to build them*
  - *mutex conditions for literals, actions*
- *How to use them*
  - *direct search, conversion to SAT*

Optimization & Search

- *Classes of optimization problem*
  - *LP, ILP, MILP*
  - *linear constraints, objective, integrality*
- *Using search for optimization*
  - *pruning w/ lower bounds on objective*
  - *stopping early w/ upper bounds*

Relaxation

- *Relaxation = increase feasible region*
- *Good way to get upper bounds on max*
- *Particularly, LP relaxation of an ILP*
- *And its dual*

Duality

- *How to find dual of an LP or ILP*
- *Interpretations of dual*
  - *linearly combine constraints to get a new constraint orthogonal to objective*
  - *find best prices for scarce resources*

Duality w/ equality

Recall duality w/ inequality

- *Take a linear combination of constraints to bound objective*
- $(a + 2b)\,w + (a + 5b)\,d \le 4a + 12b$
- $profit = 1w + 2d$
- *So, if* $1 \le (a + 2b)$ *and* $2 \le (a + 5b)$*, we know that* $profit \le 4a + 12b$

Equality example

- *minimize y subject to*
- $x + y = 1$
- $2y - z = 1$
- $x, y, z \geq 0$

Equality example

- *Want to prove bound* $y \geq \ldots$
- *Look at 2nd constraint:*

  $2y - z = 1 \quad \Rightarrow$

  $y - z/2 = 1/2$

- *Since* $z \geq 0$*, dropping* $-z/2$ *can only increase LHS* $\Rightarrow$
  - $y \geq 1/2$

Duality w/ equalities

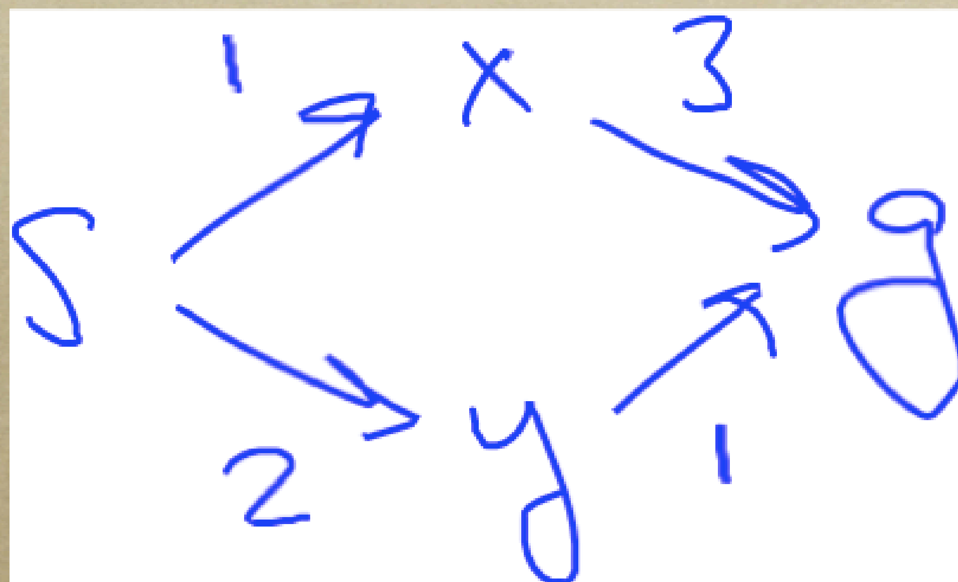
- *In general, could start from any linear combination of equality constraints*
  - *no need to restrict to +ve combination*
- $a\,(x + y - 1) + b\,(2y - z - 1) = 0$
- $a\,x + (a + 2b)\,y - b\,z = a + b$

Duality w/ equalities

- $a\,x + (a + 2b)\,y - b\,z = a + b$
- *As long as coefficients on LHS* $\leq (0, 1, 0)$,
  - *objective* $= 0\,x + 1\,y + 0\,z \geq a + b$
- *So, maximize* $a + b$ *subject to*
  - $a \leq 0$
  - $a + 2b \leq 1$
  - $-b \leq 0$

Duality example

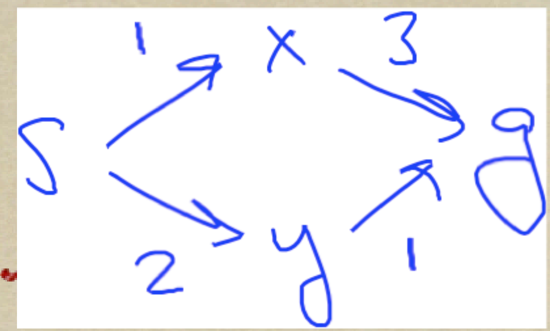
Path planning LP



- *Find the min-cost path: variables*

$$P_{sx}, P_{sy}, P_{xg}, P_{yg} \geq 0$$

Path planning LP



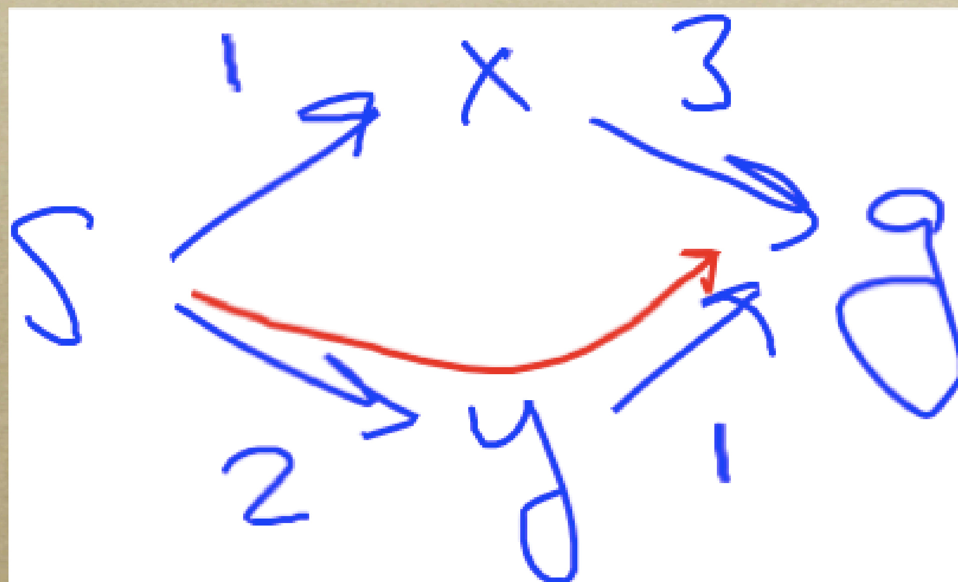
min

$$p_{sx} + 3p_{xg} + 2p_{sy} + p_{yg}$$

st

$$
\begin{aligned}
p_{sx} \quad + p_{sy} &= 1 \\
-p_{sx} + p_{xg} &= 0 \\
-p_{sy} + p_{yg} &= 0 \\
-p_{xg} - p_{yg} &= -1
\end{aligned}
$$

Optimal solution



$p_{sy} = p_{yg} = 1, \quad p_{sx} = p_{xg} = 0, \quad cost\ 3$

Matrix form

$$\min \begin{pmatrix} 1 & 3 & 2 & 1 \end{pmatrix} p$$

st

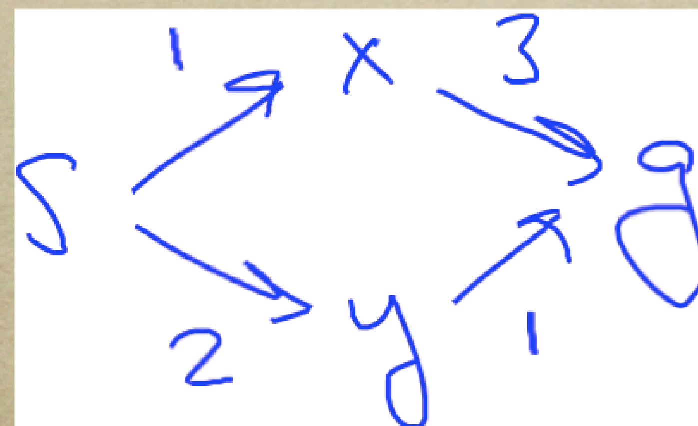
$$\begin{matrix} \lambda_s \\ \lambda_x \\ \lambda_y \\ \lambda_g \end{matrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & -1 & 0 & -1 \end{pmatrix} p = \begin{pmatrix} 1 \\ 0 \\ 0 \\ -1 \end{pmatrix}$$

$$p \geq 0$$

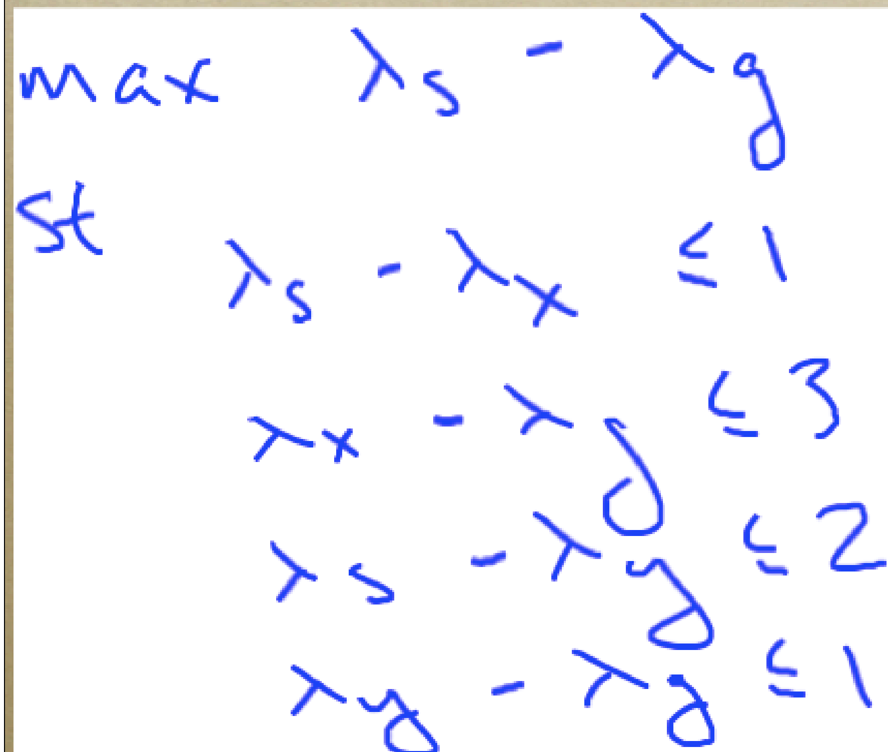
Dual

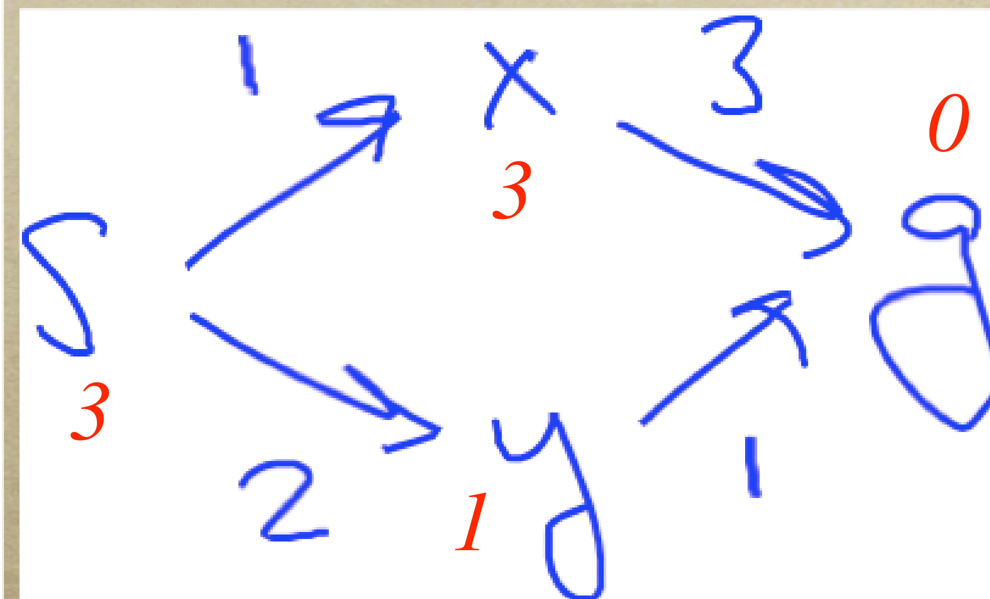
$$\max \quad \lambda_s - \lambda_g$$

$$\text{st} \quad \lambda_s - \lambda_x \le 1$$
$$\lambda_x - \lambda_g \le 3$$
$$\lambda_s - \lambda_y \le 2$$
$$\lambda_y - \lambda_g \le 1$$



Optimal dual solution





Any solution which adds a constant to all λs also works; $\lambda_x = 2$ also works

Interpretation

- *Dual variables are prices on nodes: how much does it cost to start there?*
- *Dual constraints are local price constraints: edge* xg *(cost 3) means that node x can't cost more than 3 + price of node g*

Search in ILPs

Simple search algorithm

- *Run DFS*
  - *node = partial assignment*
  - *neighbor = set one variable*
- *Prune if a constraint is unsatisfiable*
  - *E.g., in 0/1 prob, setting $y = 0$ in $x + 3y \geq 4$*
- *If we reach a feasible full assignment, calculate its value, keep best*

More pruning

- *Constraint from best solution so far: objective ≥ M (for maximization problem)*
- *Constraint from optimal dual solution: objective ≤ M*
- *Can we find more pruning to do?*

First idea

- *Analog of constraint propagation or unit resolution*
- *When we set x, check constraints w/ x in them to see if they restrict the domain of another variable y*
- *E.g., setting x to 1 in implication constraint* $(1-x) + y \geq 1$

Example

- *0/1 variables x, y, z*
- *maximize x subject to*

$$2x + 2y - z \leq 2$$

$$2x - y + z \leq 2$$

$$-x + 2y - z \leq 0$$

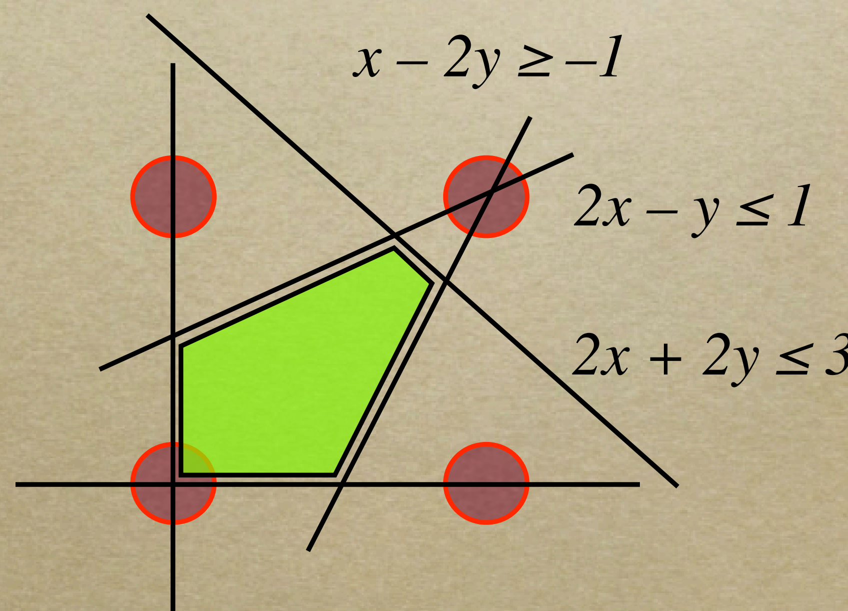
Problem w/ constraint propagation

- *Constraint propagation doesn't prune as early as it could:*

$$2x + 2y - z \leq 2$$

$$2x - y + z \leq 2$$

$$-x + 2y - z \leq 0$$



- *Consider* $z = 1$

Generalizing constraint propagation

- *Try adding two constraints, then propagating*

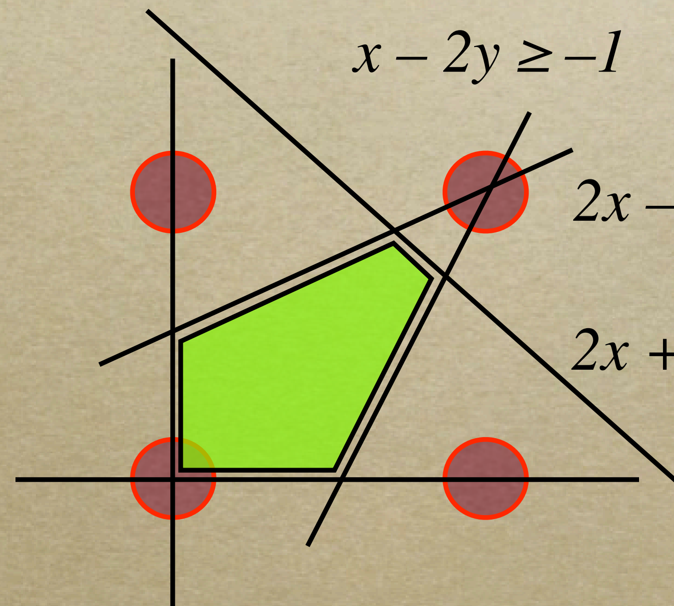
$$2x + 2y \le 3$$

$$(2x - y \le 1) * 2$$

$$\overline{\quad\quad\quad\quad\quad\quad}$$

$$6x \le 5$$

- $\Rightarrow$ *objective* $= x = 0$



Using the dual

- *We just applied the duality trick to the LP after fixing $z = 1$*
- *Used a linear combination of two constraints to get a bound on the objective*
- *Leads to an algorithm called branch and bound*

Branch and bound

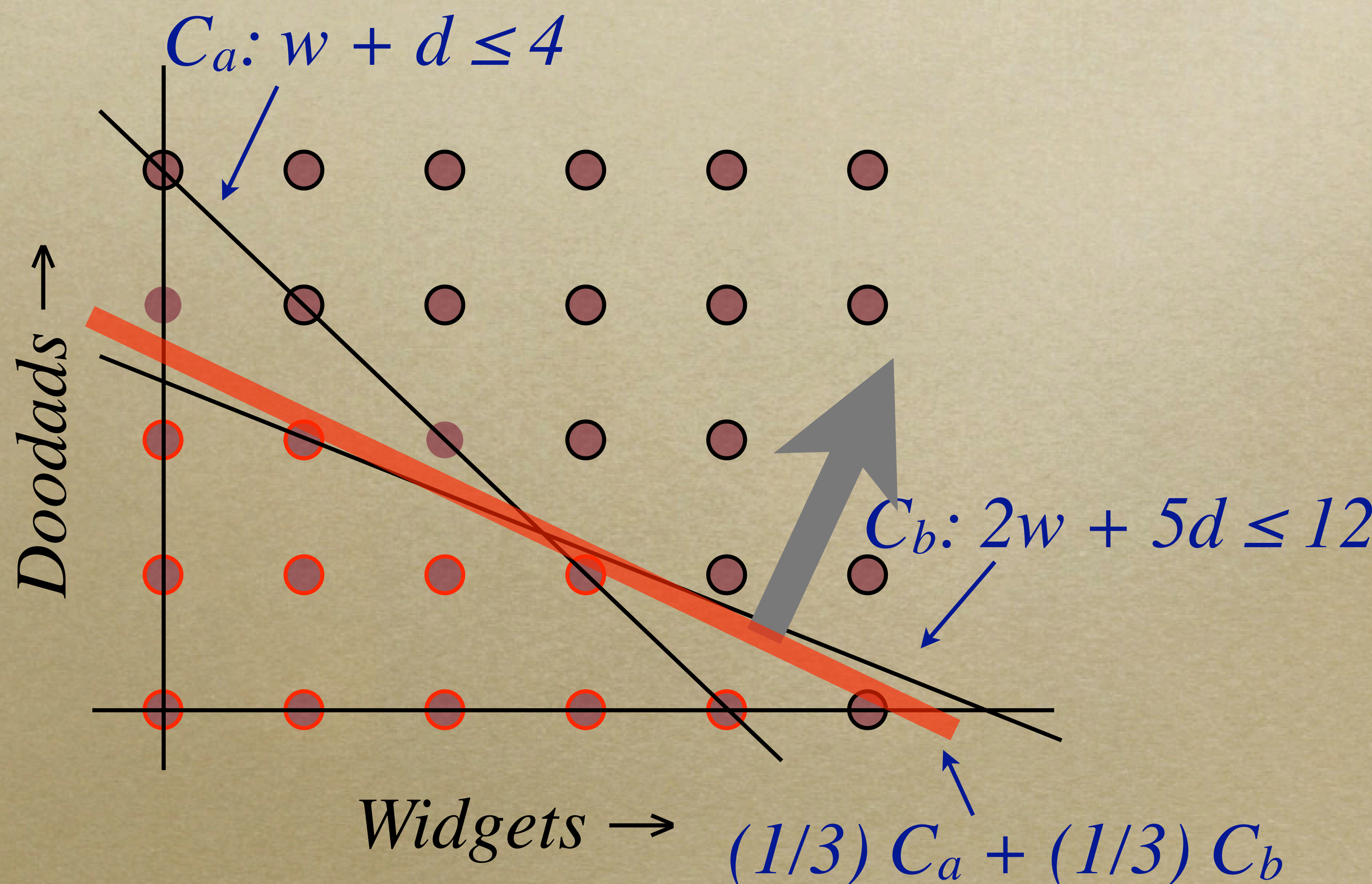
- *Each time we fix a variable, solve the resulting LP*
- *Gives a tighter upper bound on value of objective in this branch*
- *If this upper bound < value of a previous solution, we can prune*
- *Called* ***fathoming*** *the branch*

Can we do more?

- *Yes: we can make bounds tighter by looking at the…*

Duality gap

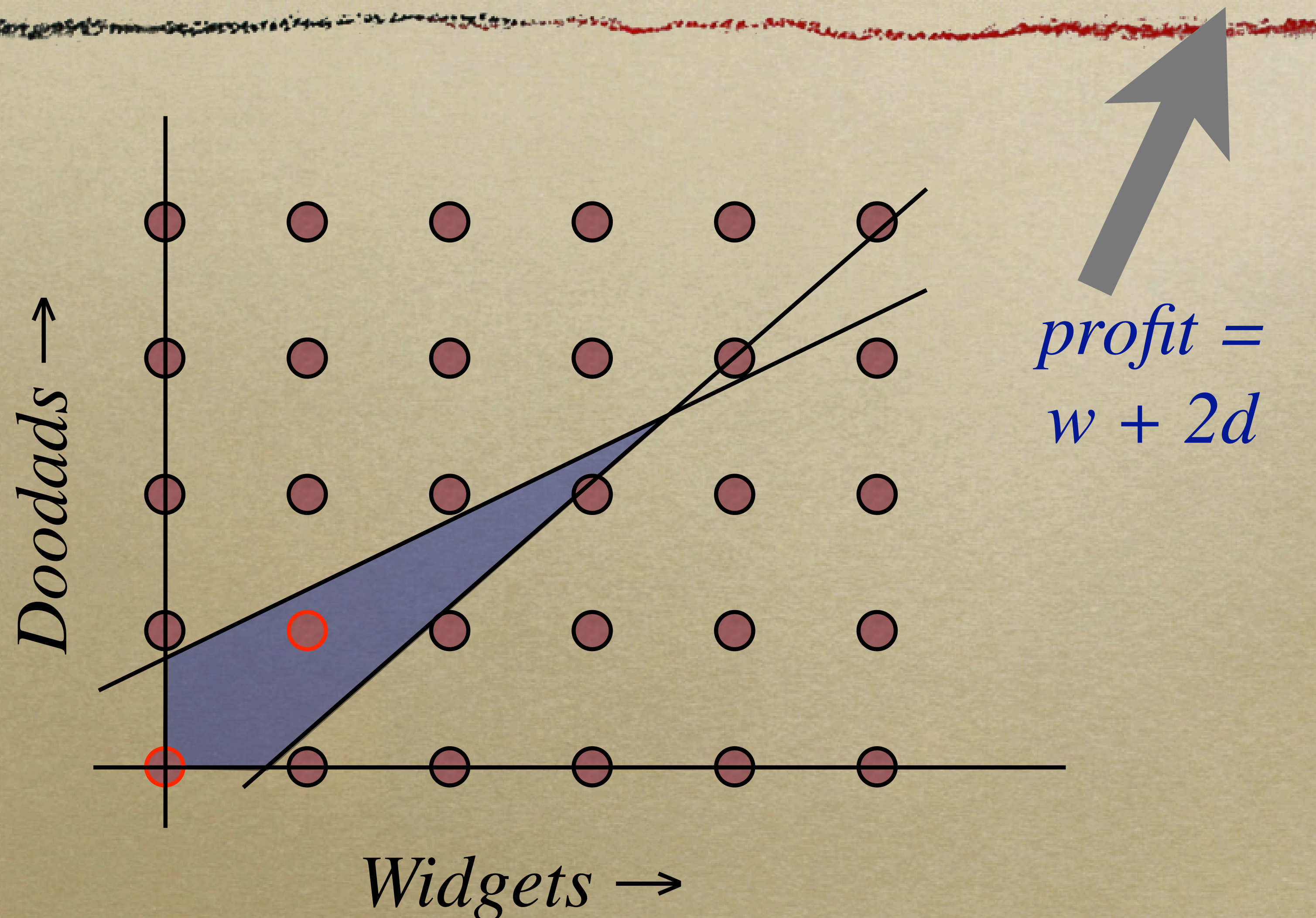
Factory LP



Duality gap

- *We got bound of 5 1/3 either from primal LP relaxation or from dual LP*
- *Compare to actual best profit of 5 (respecting integrality constraints)*
- *Difference of 1/3 is* ***duality gap***
  - *Term is also used for ratio 5 / (5 1/3)*
- *Pretty close to optimal, right?*

Unfortunately…



Bad gap

- *In this example, duality gap is 3 vs 8.5, or about a ratio of 0.35*
- *Ratio can be arbitrarily bad*

Aside: bounding the gap

- *Can often bound gap for classes of ILPs*
- *E.g., straightforward ILP from **MAX SAT***
  - *MAX SAT: satisfy as many clauses as possible in a CNF formula*
- *Gap no worse than 1–1/e = 0.632…*

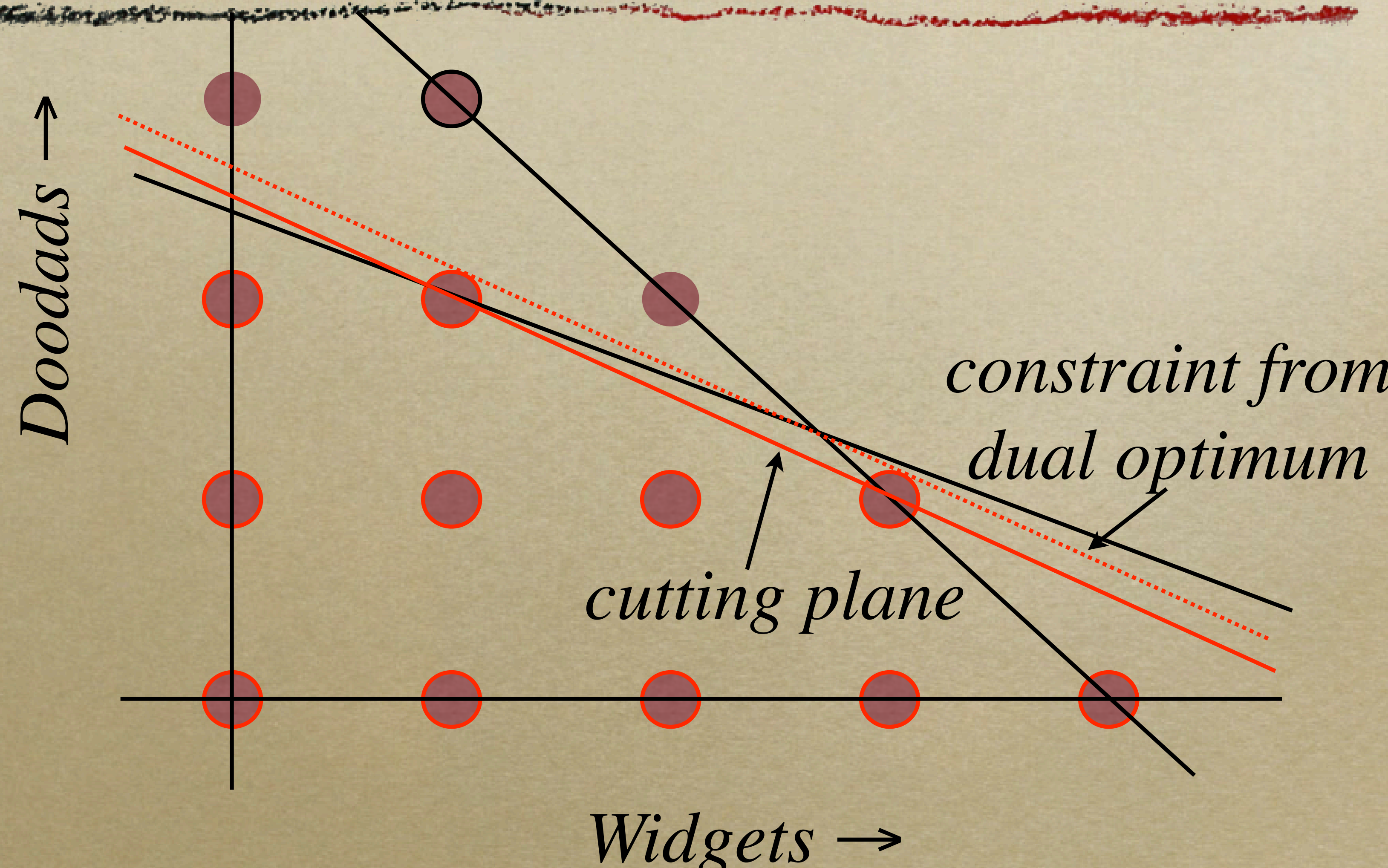
Early stopping

- *A duality gap this large won't let us prune or stop our search early*
- *To fix this problem:* ***cutting planes***

Cutting plane

- *A cutting plane is a new linear constraint that*
  - *cuts off some of the non-integral points in the LP relaxation*
  - *while leaving all integral points feasible*

Cutting plane



Cutting plane method

- *Solve the LP relaxation*
- *Use solution to find a cutting plane*
- *Add cutting plane to LP*
  - *LP is now a stronger relaxation*
- *Repeat*
  - *until solution to LP is integral*

How can we find a cutting plane?

- *One suggestion: Gomory cuts*
  - *R. E. Gomory, 1963*
- *First to guarantee finite termination of cutting plane method*
- *Example above was a Gomory cut*

Gomory cut example

- *A linear combination of constraints:*

  $w + 2d \leq 5\ 1/3$

- *Since w, d are integers, so is w + 2d*
- *So we also have*

  $w + 2d \leq 5$

- *Can (but won't) generalize recipe*

Cutting planes

- *How good is the Gomory cut in general?*
- *Sadly, not so great.*
- *Other general cuts have been proposed, but best cuts are often problem-specific*

Branch and Cut

Branch and cut

- *Cutting planes recipe doesn't use branching*
- *What if we try to interleave search with cut generation?*
- *Resulting **branch and cut** methods are some of the most popular algorithms for solving ILPs and MILPs*

Recipe

- *DFS as for branch and bound*
- *Every so often, solve LP relaxation*
  - *prune if bound shows branch useless*
  - *while not bored, use solution to generate cut, re-solve*
- *Branch on next variable, repeat*

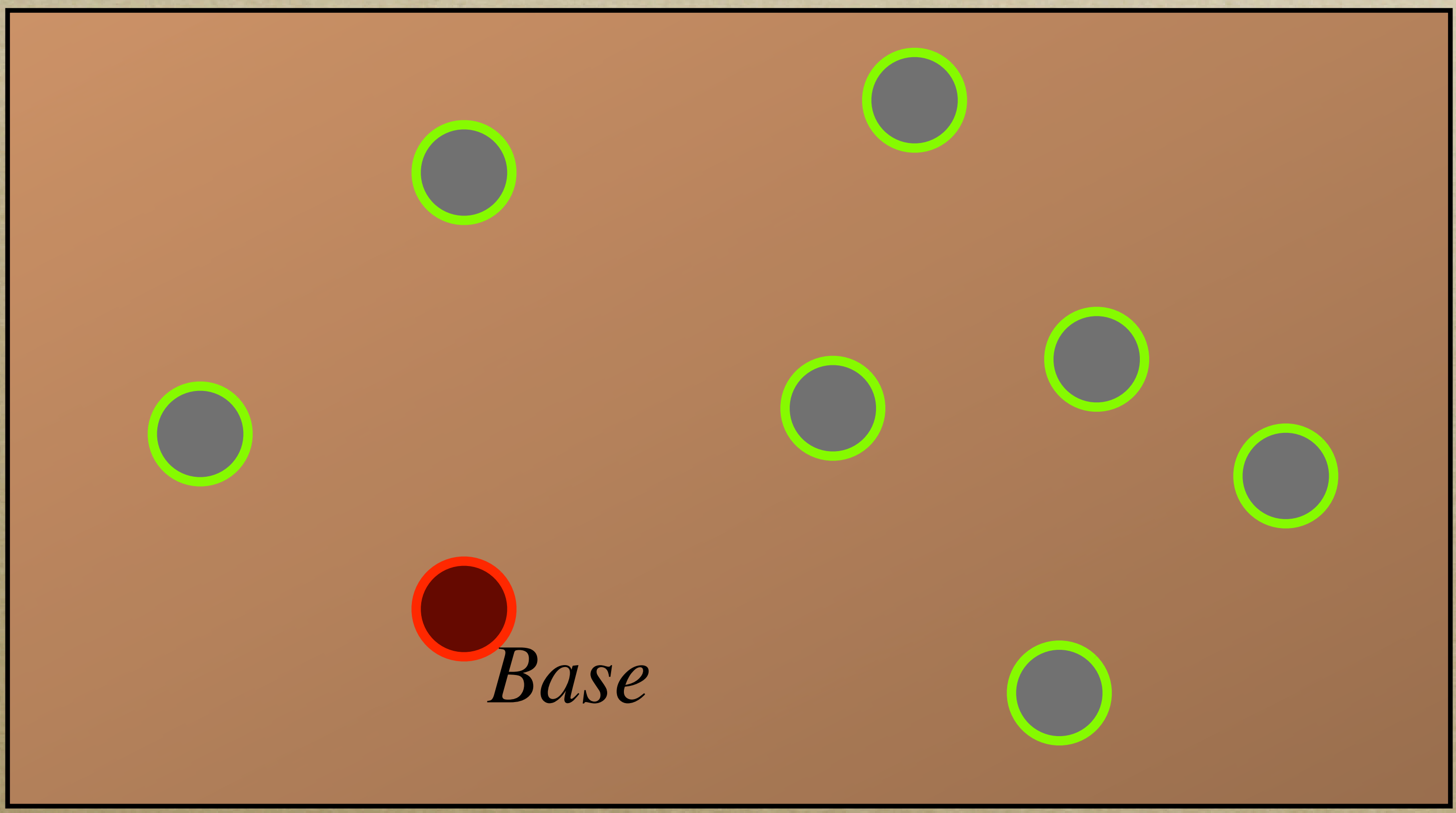
Tension of cutting v. branching

- *After a branch it may become easier to generate more cuts*
  - *so easier as we go down the tree*
- *Cuts at a node N are valid at N's children*
  - *so it's worth spending more effort higher in the search tree*

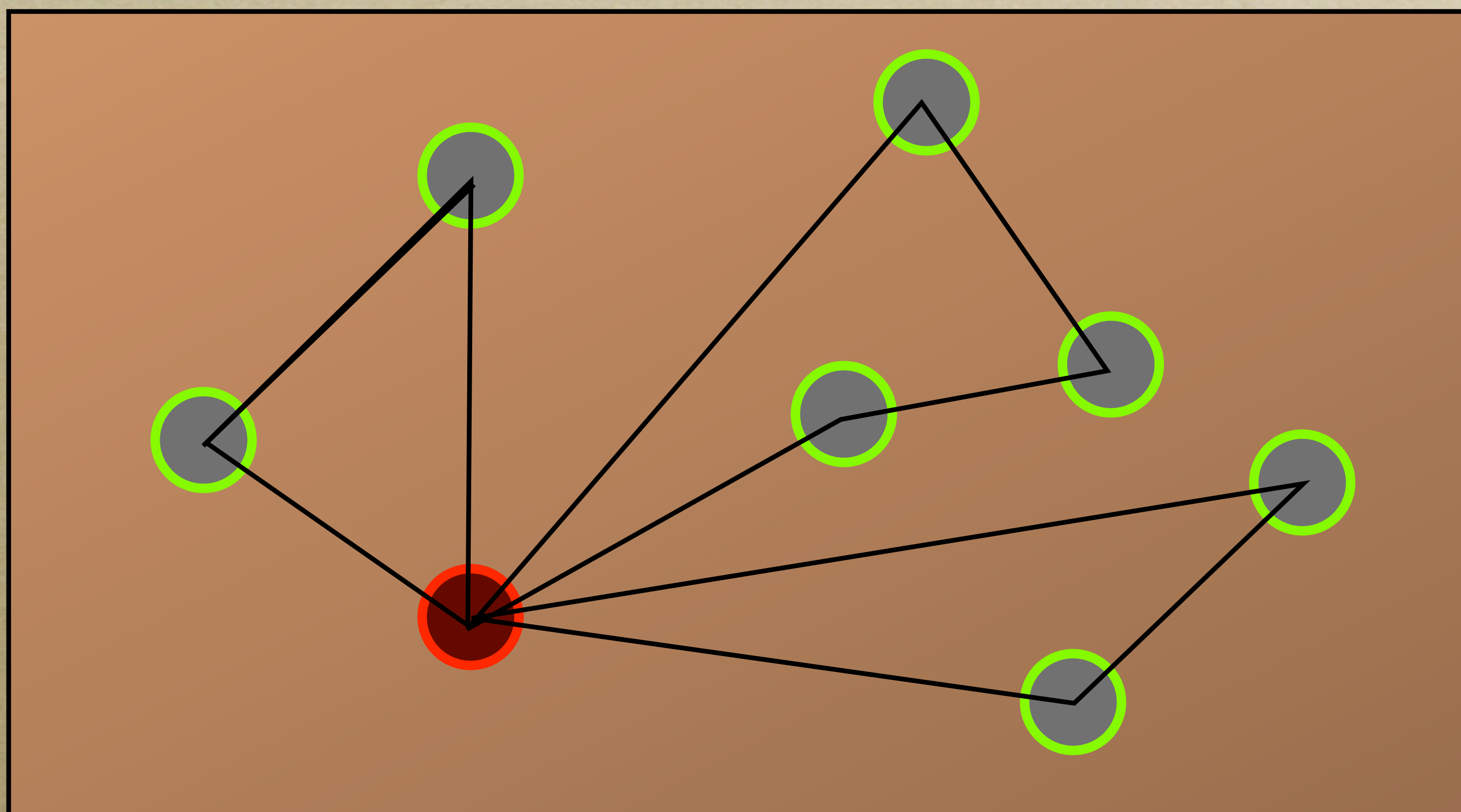
Example: robot task assignment

- *Team of robots must explore unknown area*

Points of interest



Exploration plan



ILP

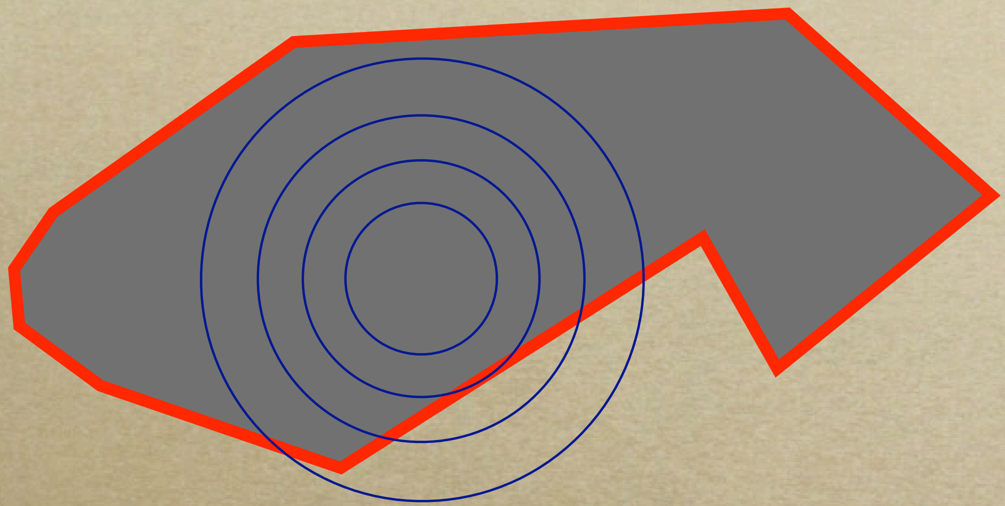
- *Variables (all 0/1):*
  - z_{ij} *= robot i does task j*
  - x_{ijkt} *= robot i uses edge jk at step t*
- *Cost = path cost – task bonus*
  - $\sum x_{ijkt} c_{ijkt} - \sum z_{ij} t_{ij}$

Constraints

- *Assigned tasks:* $\forall i, j, \sum_{kt} x_{ikjt} \geq z_{ij}$
- *One edge per step:* $\forall i, t, \sum_{jk} x_{ijkt} = 1$
  - *self-loops @ base to allow idling*
- *For each i,* x_{ijkt} *forms a tour from base:*
  - $\forall i, j, t, \sum_k x_{ikjt} = \sum_k x_{ijk(t+1)}$
  - *edges used into node = edges used out*

More on duality, search, optimization

General optimization



- *minimize f(x) over region defined by pieces*

  $g_i(x) = 0$ *or* $g_i(x) \leq 0$

- *assume f(x) convex, so difficulty is g*

Minimization

- *Unconstrained: set* $\nabla f(x) = 0$
- *E.g., minimize*

$$f(x, y) = x^2 + y^2 + 6x - 4y + 5$$

$$\nabla f(x, y) = (2x + 6, 2y - 4)$$

$$(x, y) = (-3, 2)$$

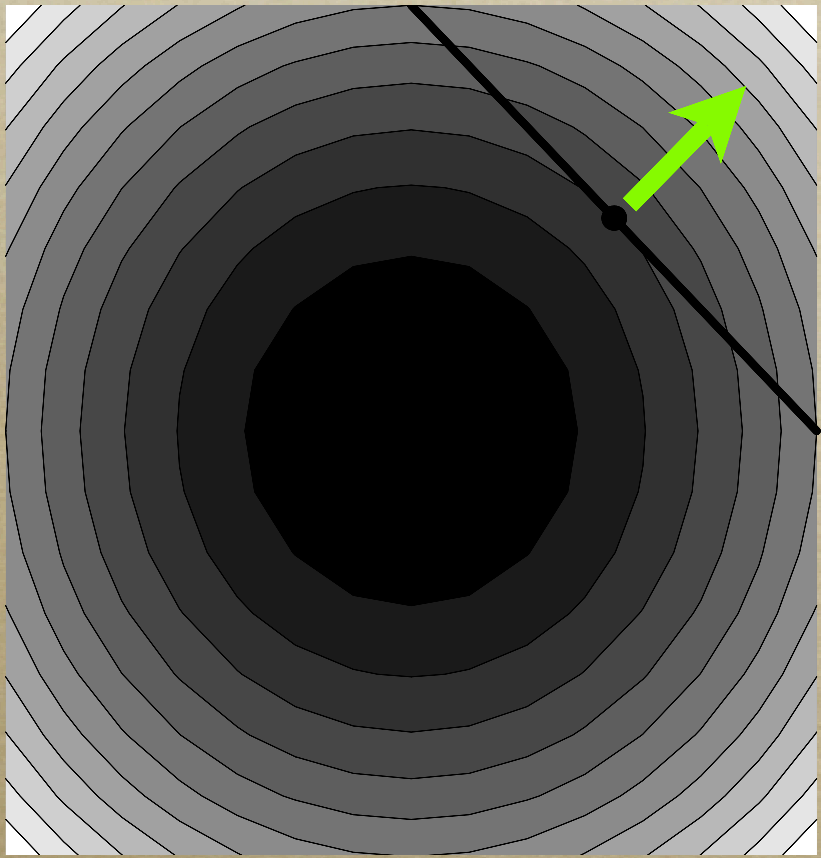
Equality constraints

- *Equality constraint:*

  minimize $f(x)$ *s.t.* $g(x) = 0$

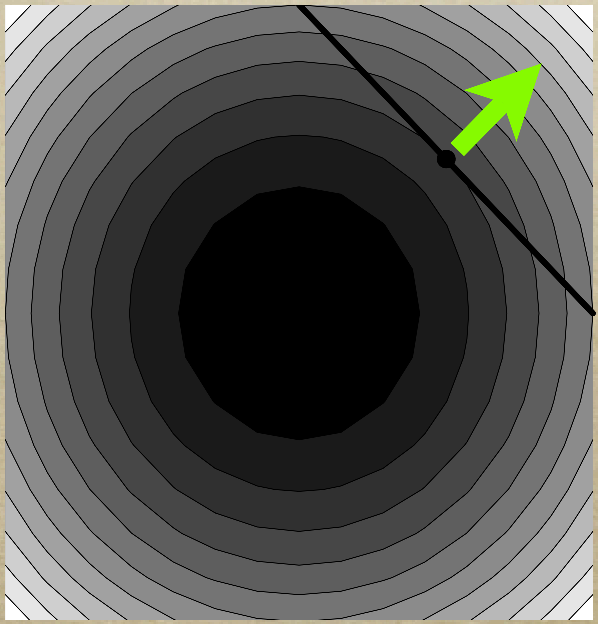
- *can't just set* $\nabla f = 0$ *(might violate* $g(x) = 0$*)*
- *Instead, objective gradient should be along constraint normal*
  - *any motion that decreases objective will violate the constraint*

Example



- *Minimize* $x^2 + y^2$ *subject to* $x + y = 2$

Lagrange multipliers



- *Minimize f(x) s.t. g(x) = 0*
- *Constraint normal is* ∇g
  - *(1, 1) in our example*
- *Want* ∇f *parallel to* ∇g
- *Equivalently, want* $\nabla f = \lambda \nabla g$
- λ *is a* ***Lagrange multiplier***

Lagrange multipliers

- *Original constraint:* $x + y = 2$
- $\nabla f = \lambda \nabla g$: $(2x, 2y) = \lambda(1, 1)$

$$x + y = 2$$

$$2x = \lambda$$

$$2y = \lambda$$

More than one constraint

- *With multiple constraints, use multiple multipliers:*

$$\min x^2 + y^2 + z^2 \text{ st}$$

$$x + y = 2$$

$$x + z = 2$$

$$(2x, 2y, 2z) = \lambda(1, 1, 0) + \mu(1, 0, 1)$$

5 equations, 5 unknowns

$$x + y = 2$$

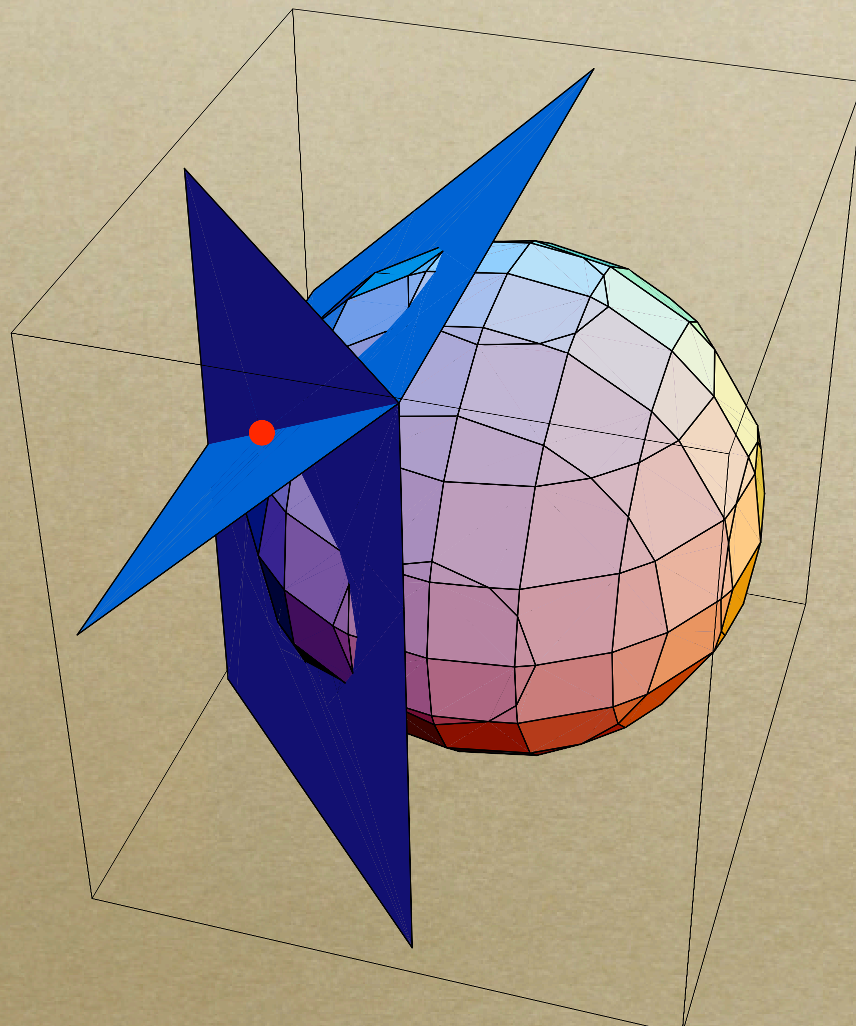
$$x + z = 2$$

$$2x = \lambda + \mu$$

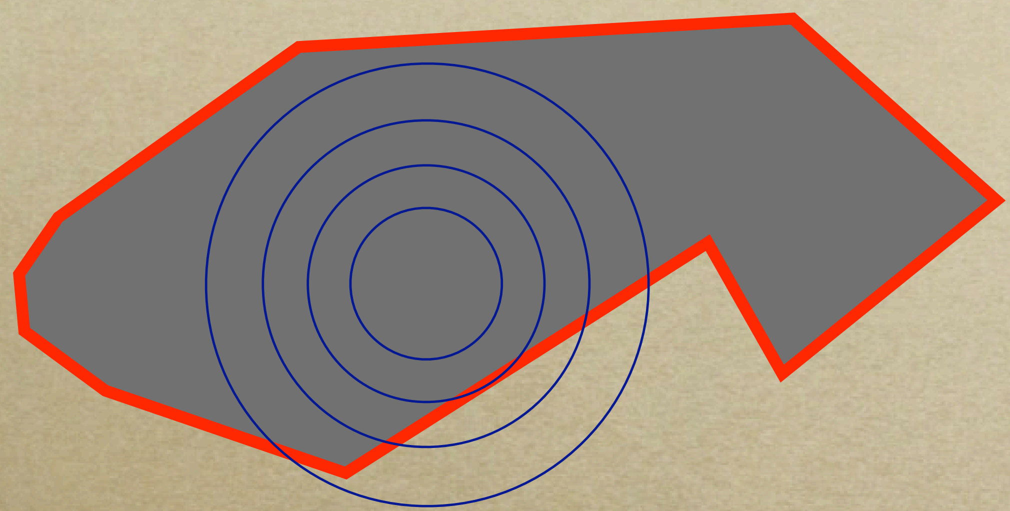
$$2y = \lambda$$

$$2z = \mu$$

Two constraints: the picture

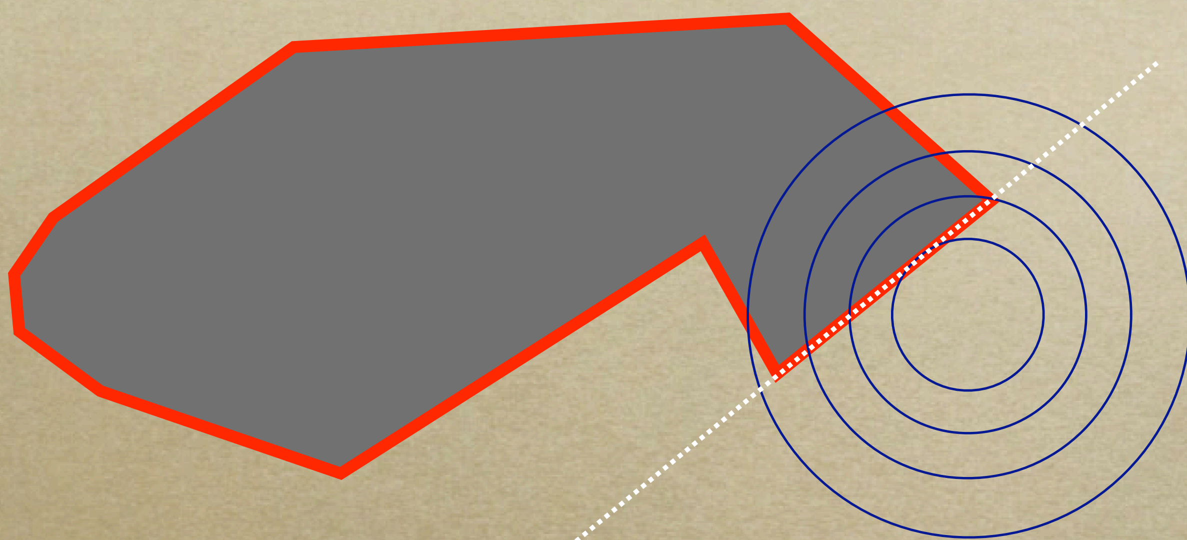


What about inequalities?



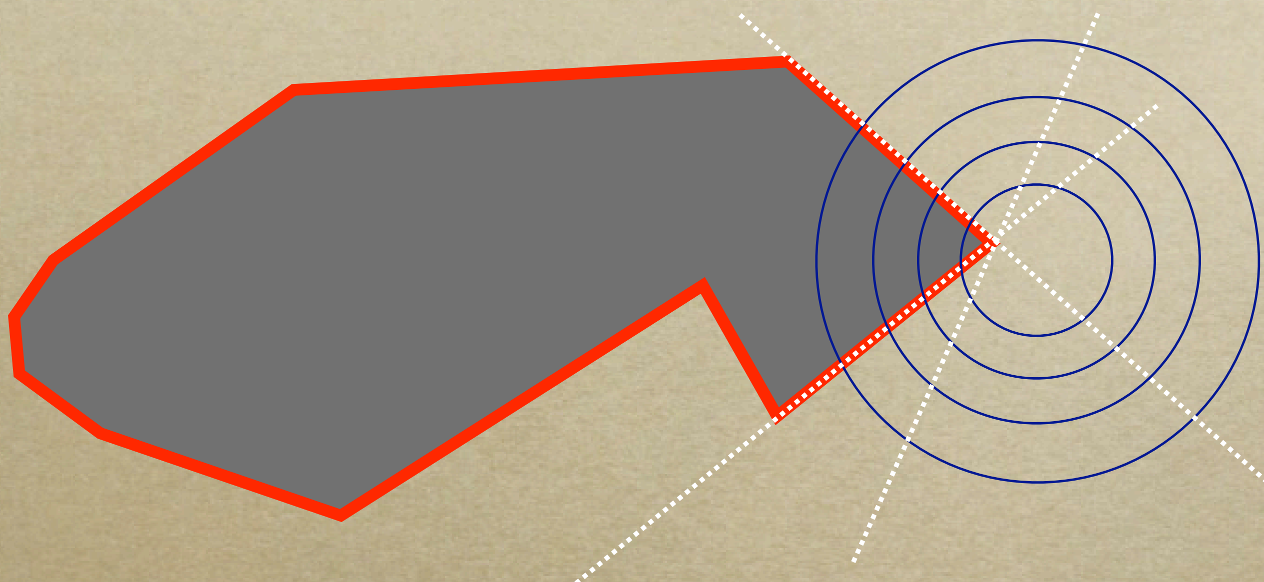
- *If minimum is in interior, can get it by setting* $\nabla f = 0$

What about inequalities?



- *If minimum is on boundary, treat as if boundary were an equality constraint (use Lagrange multiplier)*

What about inequalities?



- *Minimum could be at a corner: two boundary constraints active*
- *In n dims, up to n linear inequalities may be active (more if degenerate)*

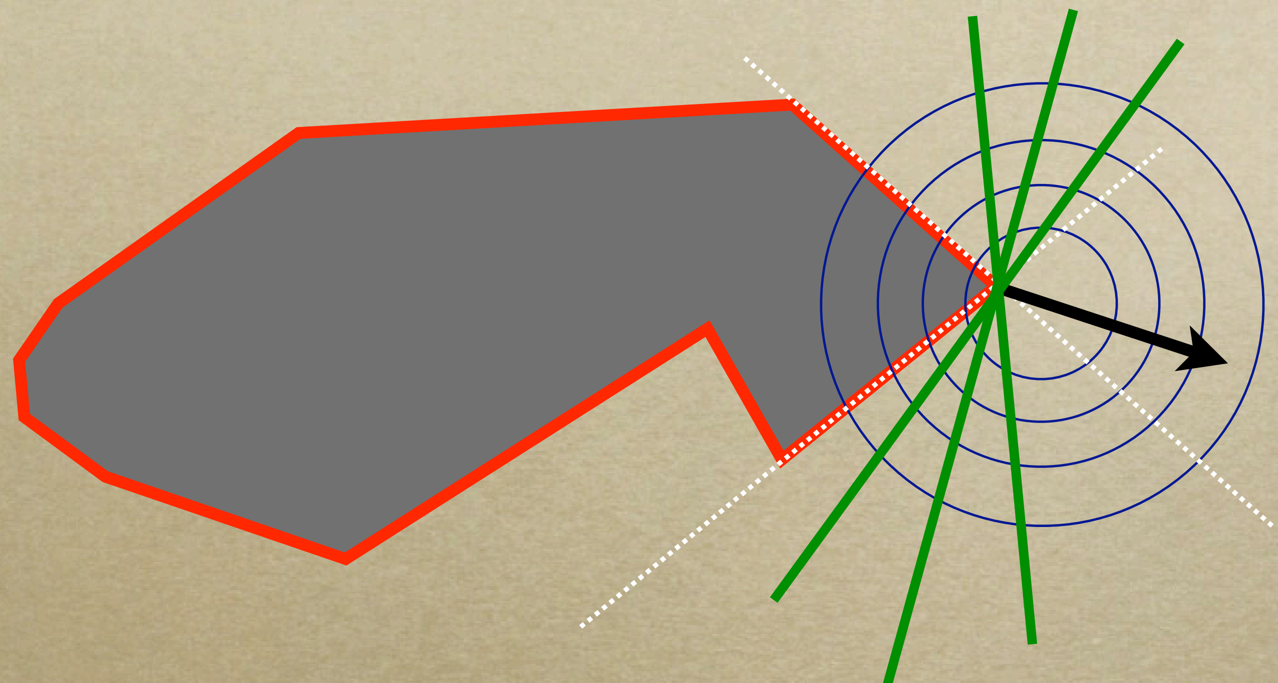
Search

- *So, a strategy for solving problems with inequality constraints: search through sets of constraints that might be active*
- *For each active set, solve linear system of equations, get a possible solution*
- *Test whether solution is feasible*
- *If so, record objective value*

Search

- *Search space:*
  - *node = active set of constraints*
    - *corresponds to a setting of variables (solve linear system)*
  - *objective = as given, plus penalty for constraint violations*
  - *neighbor = add, delete, or swap constraints*

Connection to duality



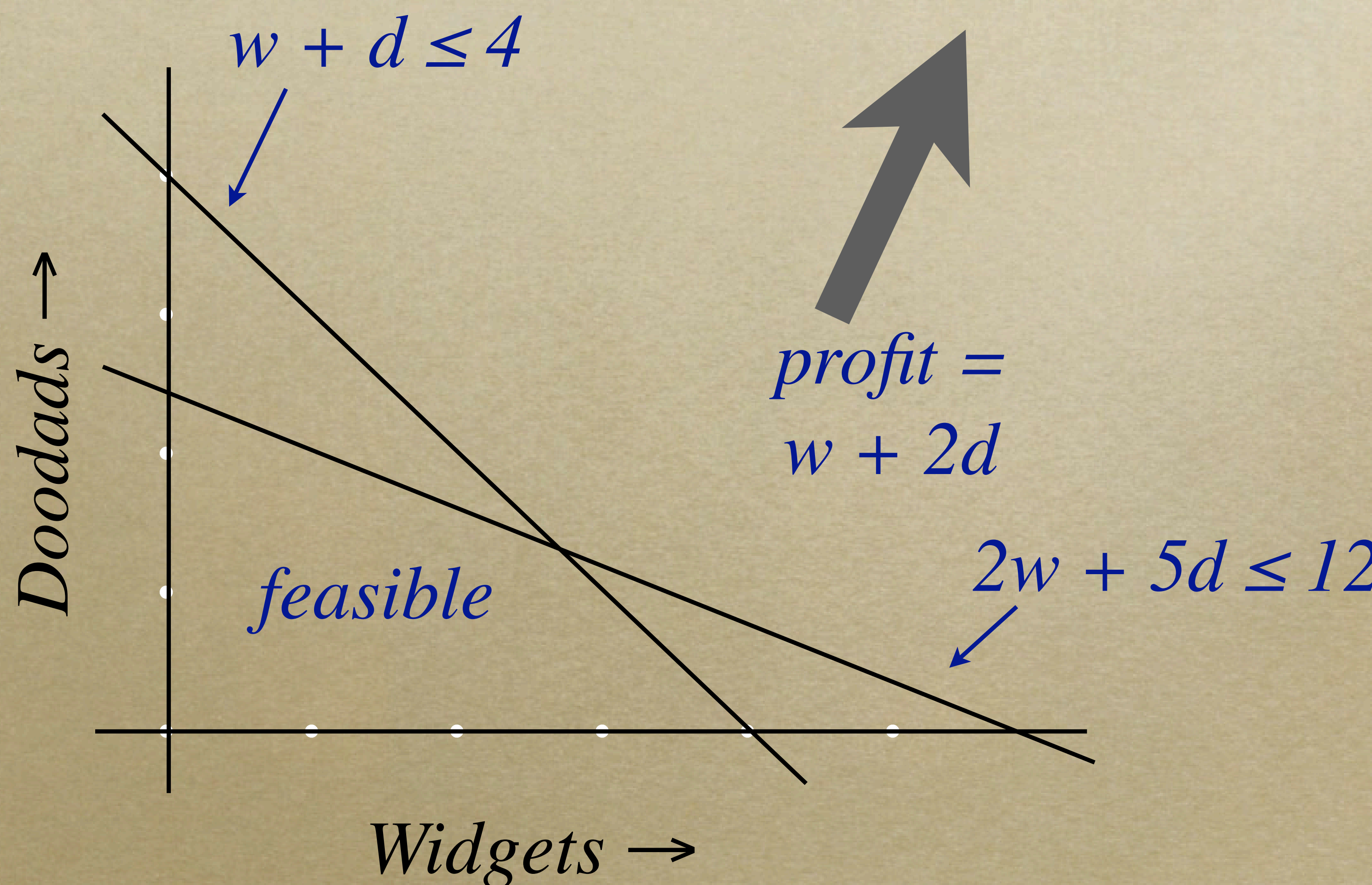
- *Linear combination of constraint normals = gradient of objective*

Connection to duality

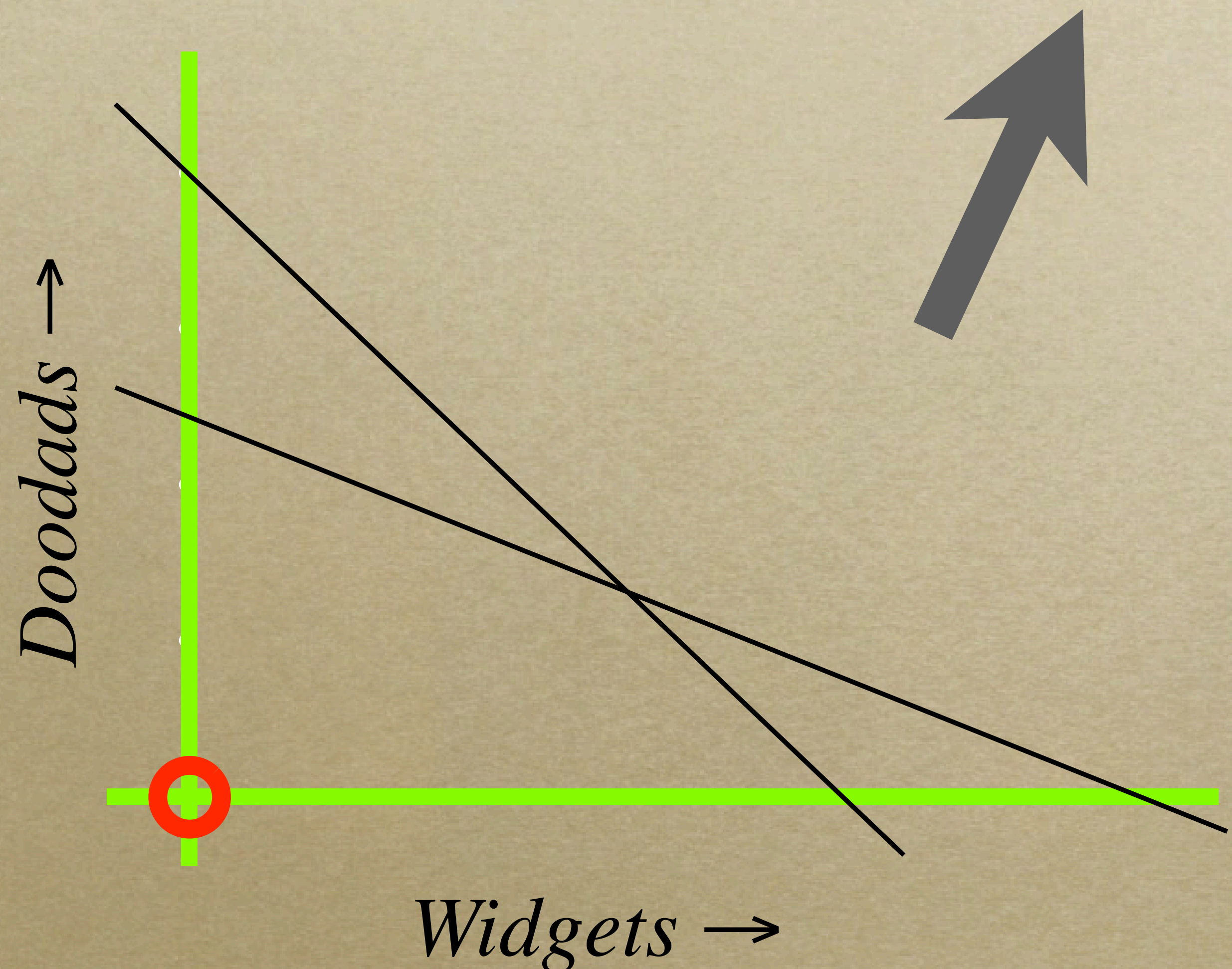
- *Each active set defines Lagrange multipliers* λ
  - *active set* $G(x) = 0$
  - $\nabla f = \nabla G \, \lambda$
- *Multipliers at optimal solution are optimal dual solution*

LPs and Simplex

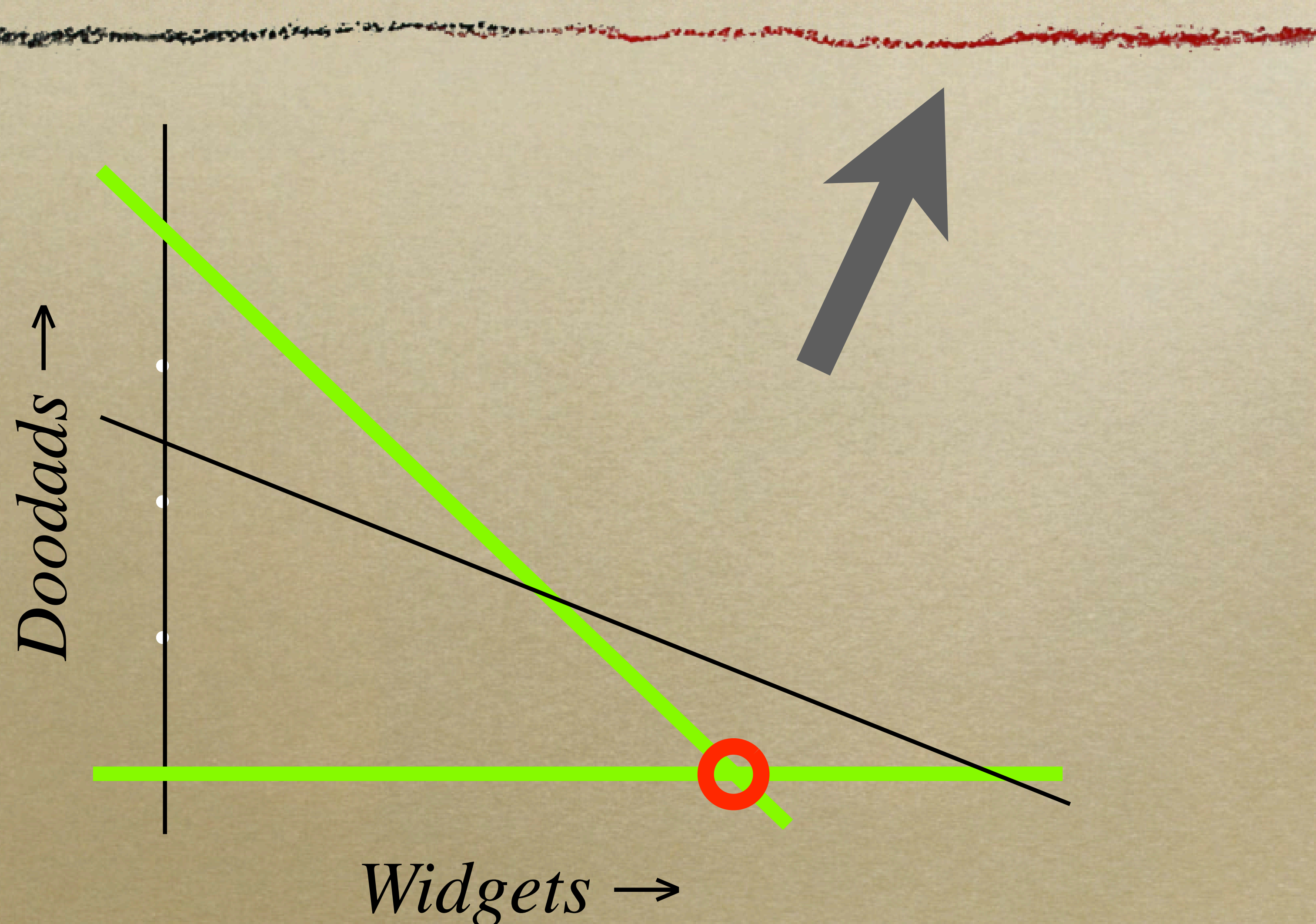
Back to LP



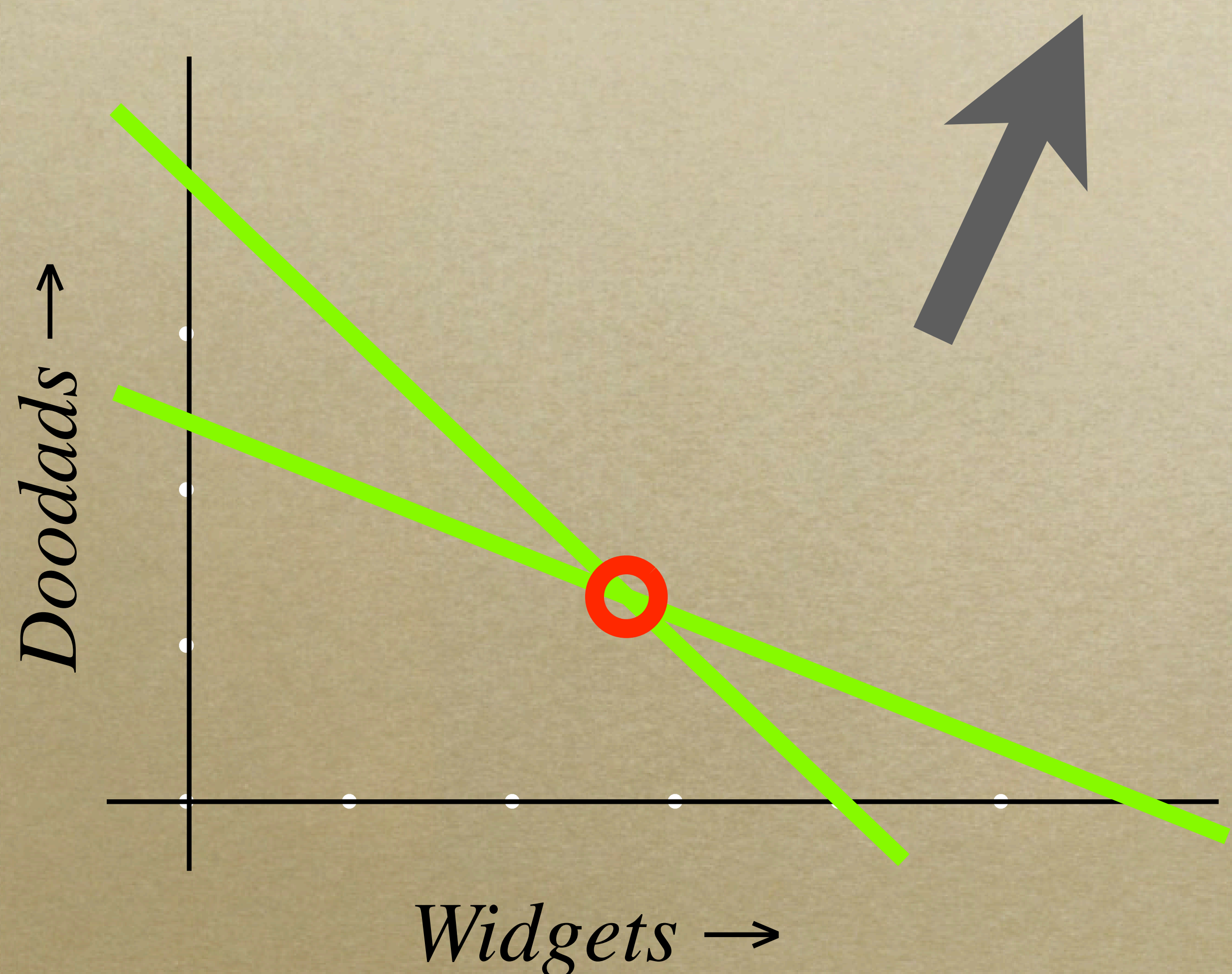
Back to LP



Back to LP



Back to LP



Simplex

- *Objective increased monotonically throughout search*
- *Turns out, this is always possible—leads to a lot of pruning!*
- *We have just defined the simplex algorithm*

Connection to duality

- *Each active set defines Lagrange multipliers*
  - *min c'x s.t. Ax = b   (A, b = active set)*
  - $\nabla(c'x) = c$
  - $\nabla(Ax - b) = A'$
  - *So,* $A'\lambda = c$
- *Multipliers at optimal solution are optimal dual solution*

Game search

Games

- *We will consider games like checkers and chess:*
  - *sequential*
  - *zero-sum*
  - *deterministic, alternating moves*
  - *complete information*
- *Generalizations later*

Chess

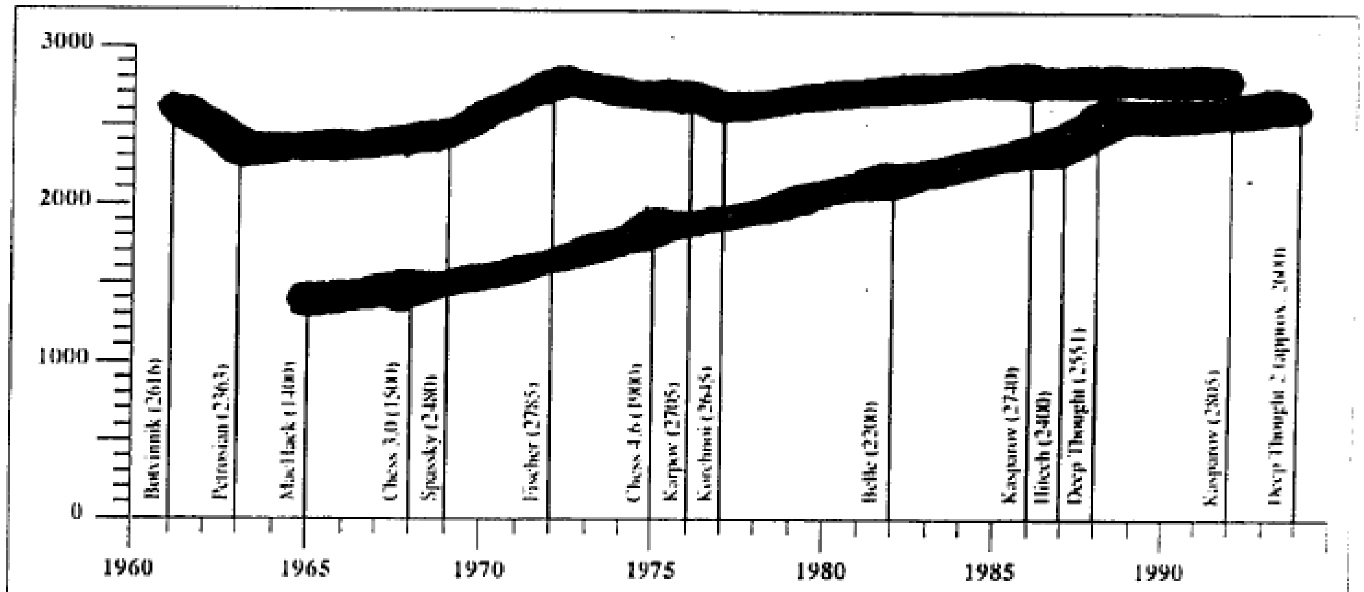


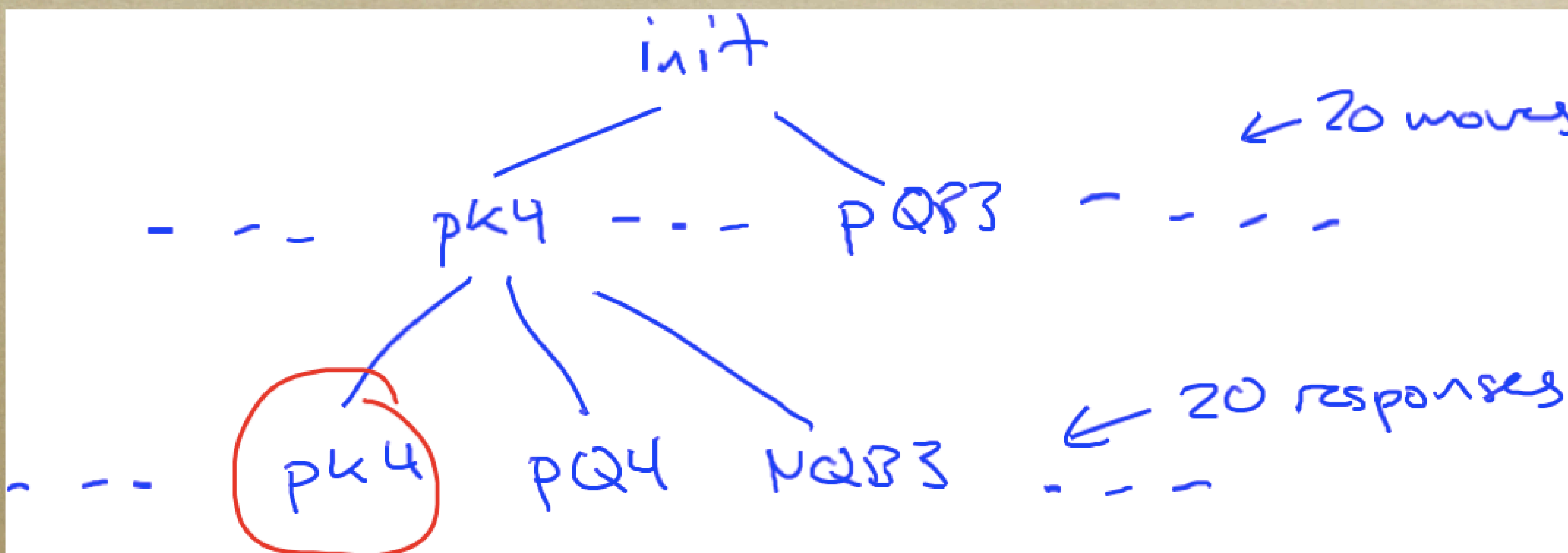
Figure 5.12 Ratings of human and machine chess champions.

- *Classic AI challenge problem*
- *In late '90s, Deep Blue became first computer to beat reigning human champion*

History:

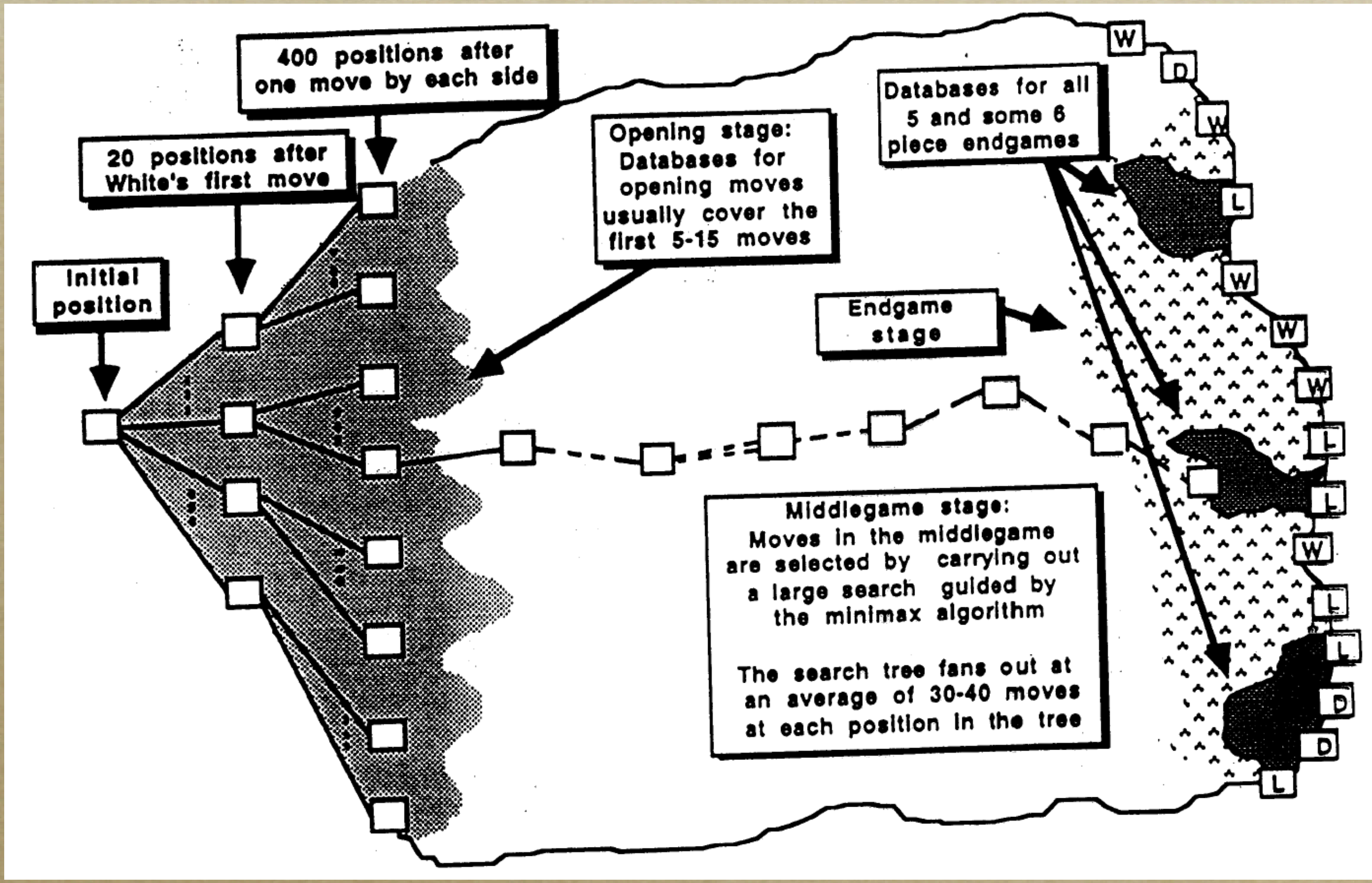
- *Minimax with heuristic: 1950*
- *Learning the heuristic: 1950s (Samuels' checkers)*
- *Alpha-beta pruning: 1966*
- *Transposition tables: 1967 (hash table to find dups)*
- *Quiescence: 1960s*
- *DFID: 1975*
- *End-game databases: 1977 (all 5-piece and some 6)*
- *Opening books: ?*

Game tree




8
7
6
5
4
3
2
1
a b c d e f g h

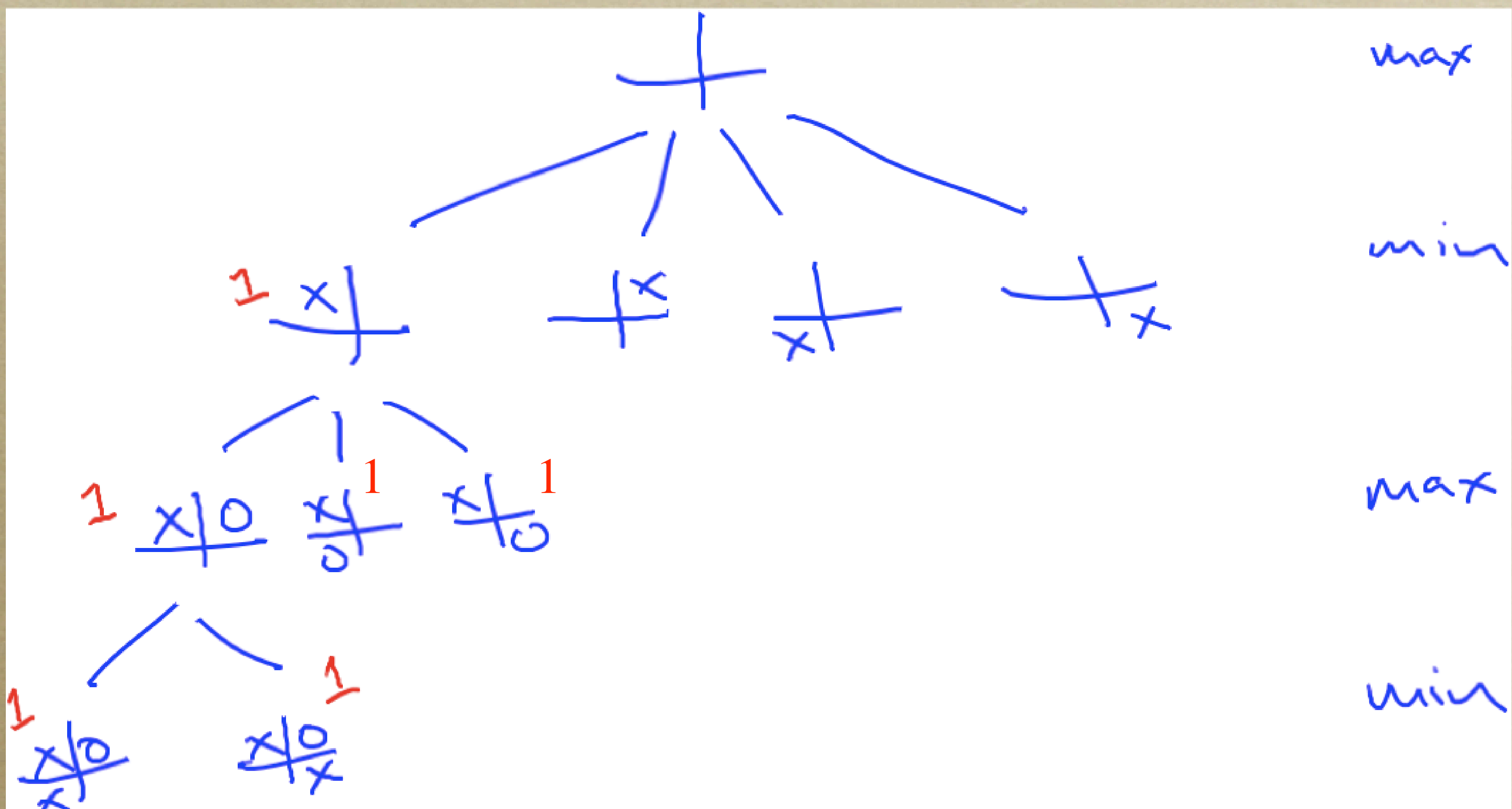
Game tree for chess



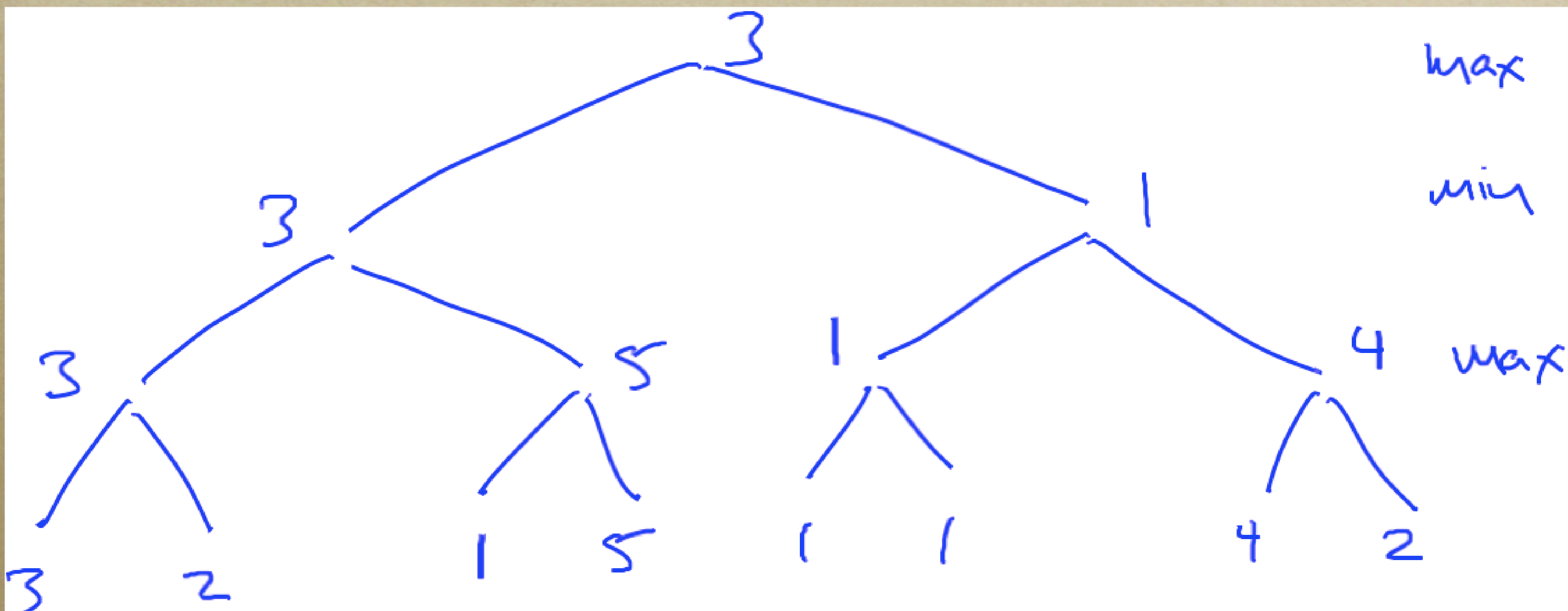
Minimax search

- *For small games, we can determine the value of each node in the game tree by working backwards from the leaves*
- *My move: node’s value is maximum over children*
- *Opponent move: value is minimum over children*

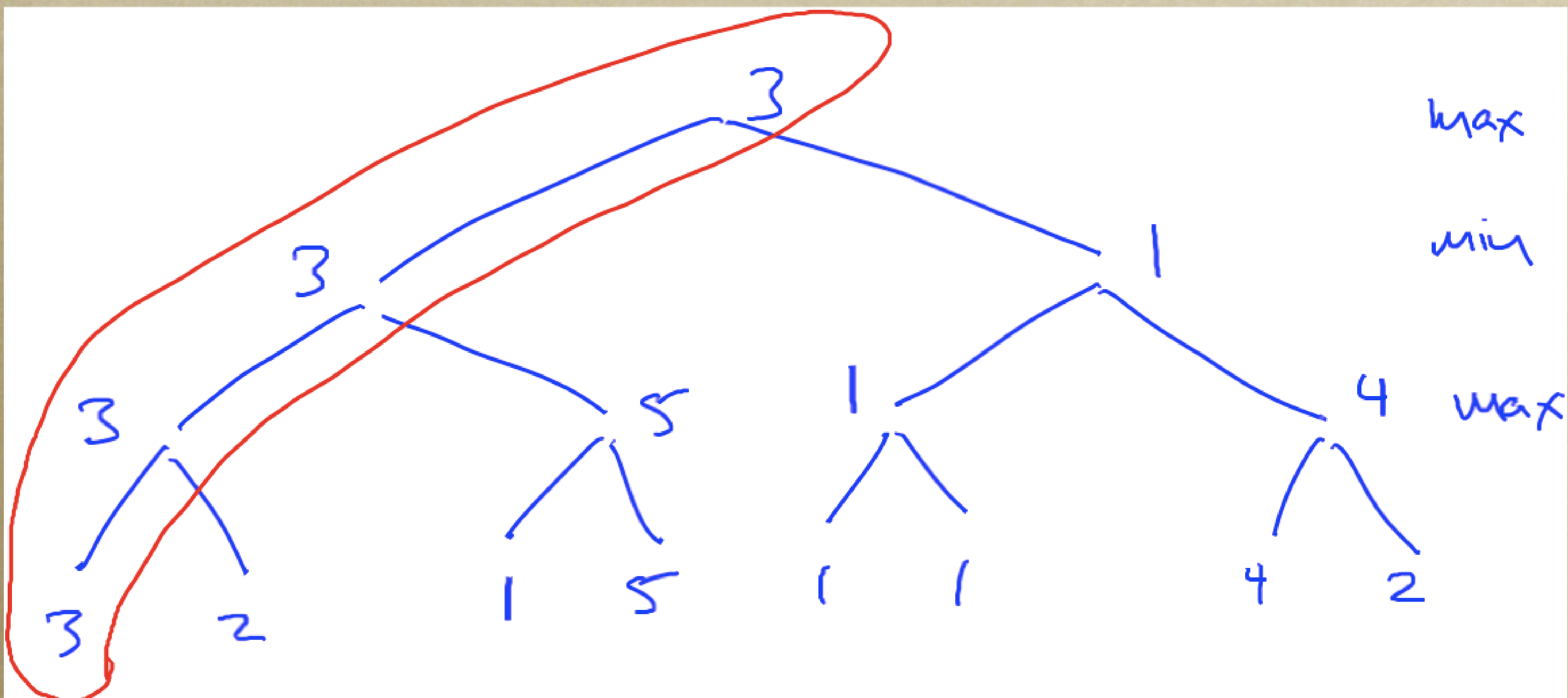
Minimax example: 2x2 tic-tac-toe



Synthetic example



Principal variation



Making it work

- *Minimax is all well and good for small games*
- *But what about bigger ones? 2 answers:*
  - *cutting off search early (big win)*
  - *pruning (smaller win but still useful)*

Heuristics

- *Quickly and approximately evaluate a position without search*
- *E.g., Q = 9, R = 5, B = N = 3, P = 1*
- *Build out game tree as far as we can, use heuristic at leaves in lieu of real value*
  - *might want to build it out unevenly (more below)*

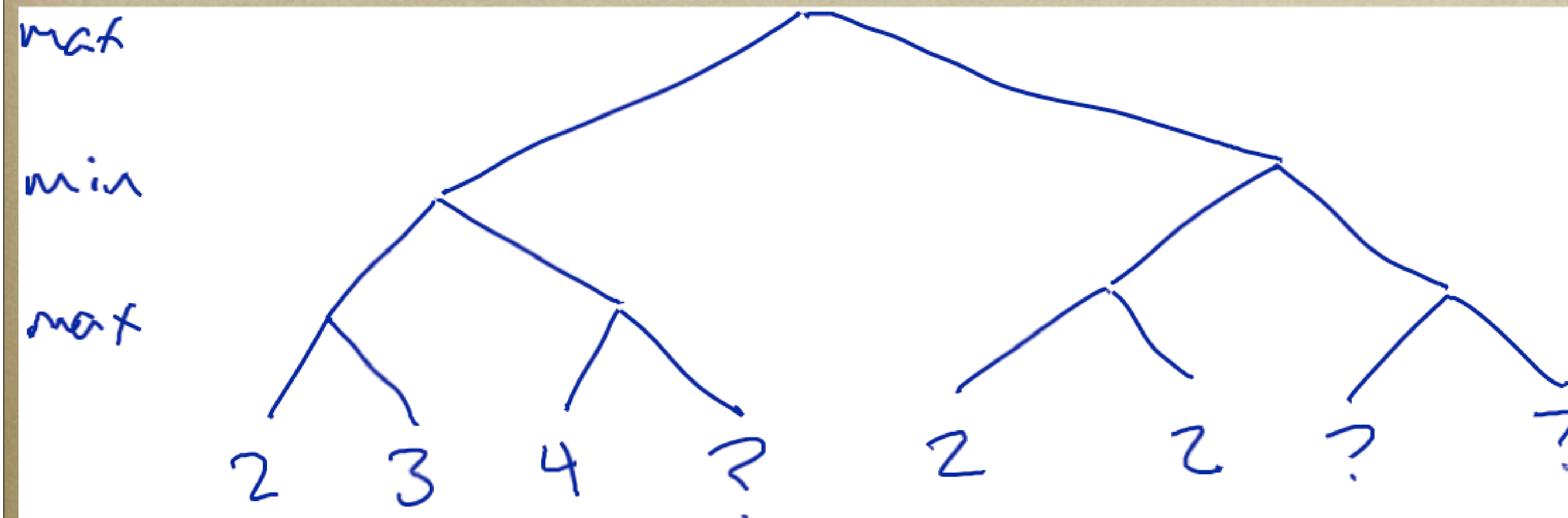
Heuristics

- *Deep Blue used: materiel, mobility, king position, center control, open file for rook, paired bishops/rooks, … (> 6000 total features!)*
- *Weights are context dependent, learned from DB of grandmaster games then hand tweaked*

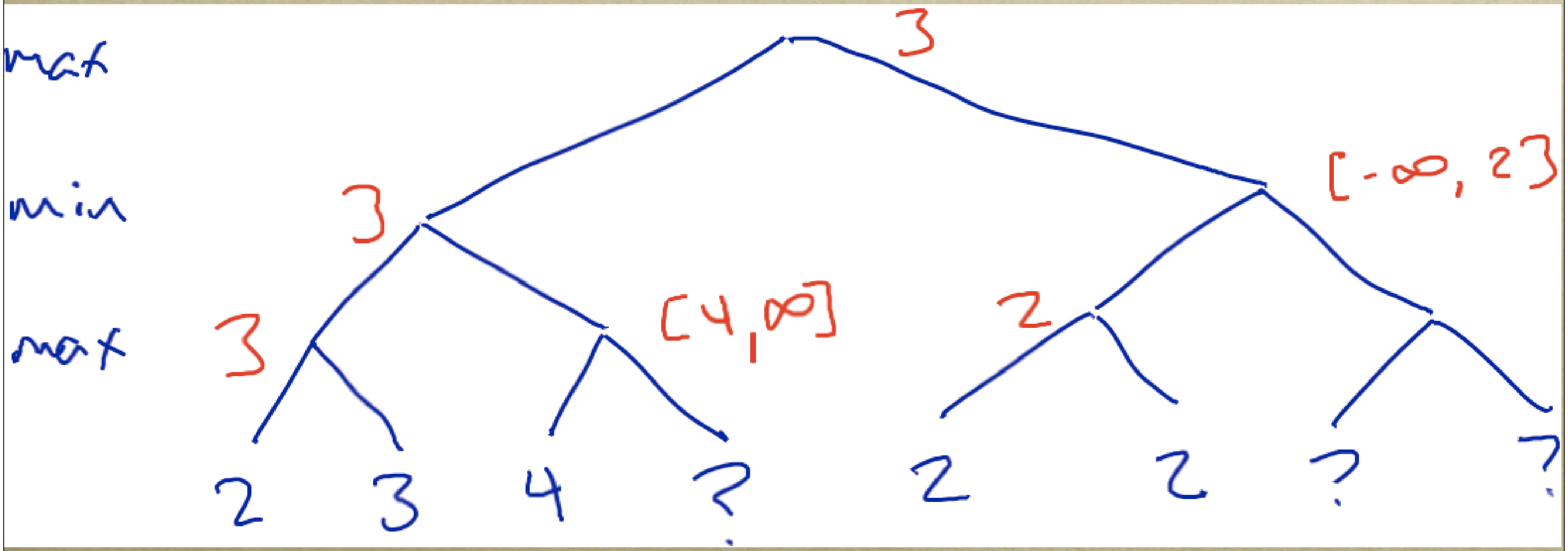
Pruning

- *Idea: don't bother looking at parts of the tree we can prove are irrelevant*

Pruning example



Pruning example



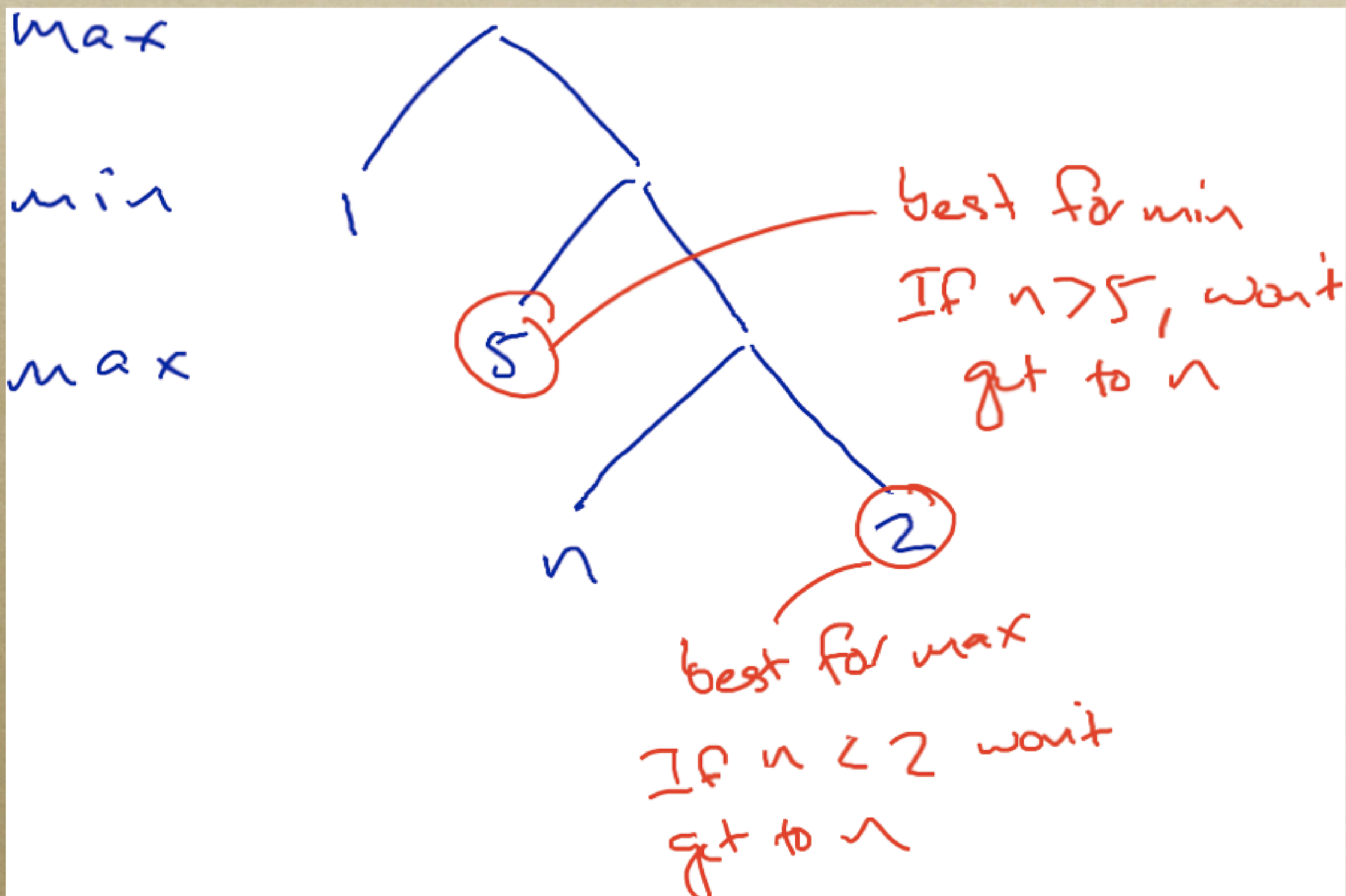
Alpha-beta pruning

- *Do a DFS through game tree*
- *At each node n on stack, keep bounds*
  - α*(n): value of best deviation so far for MAX along path to n*
  - β*(n): value of best deviation so far for MIN along path to n*

Alpha-beta pruning

- *Deviation = way of leaving the path to n*
- *So, to get* α,
  - *take all MAX nodes on path to n*
  - *look at all their children that we've finished evaluating*
  - *best (highest) of these children is* α
- *Lowest of children of MIN nodes is* β

Example of alpha and beta



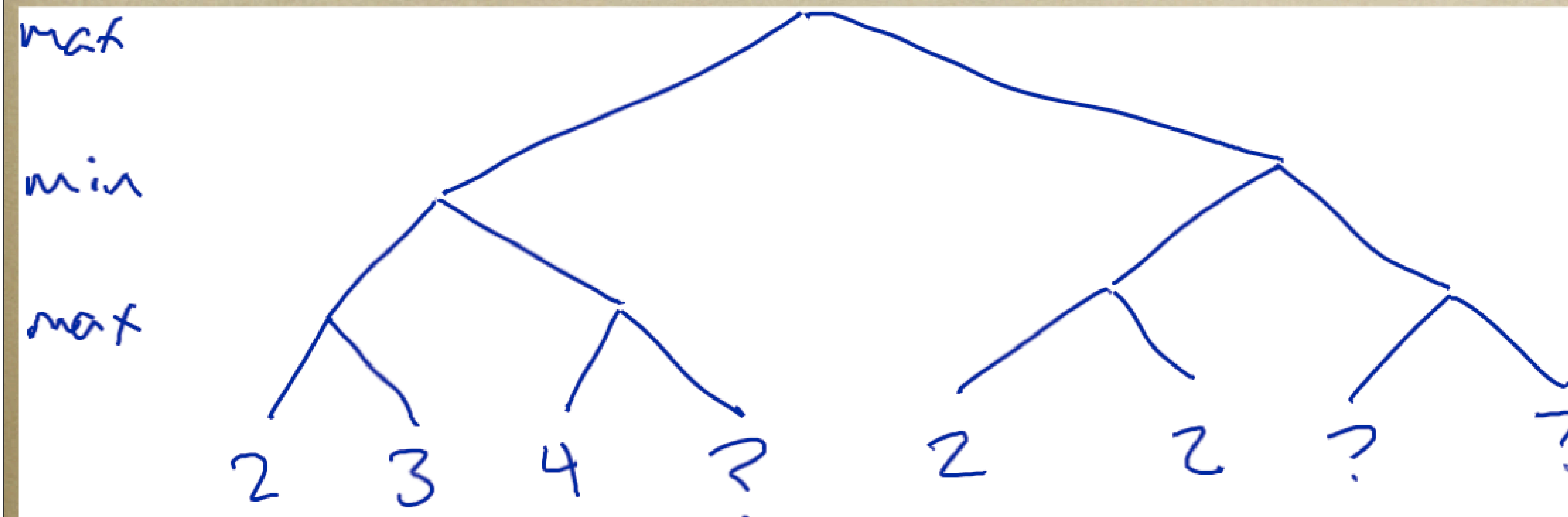
Alpha-beta pruning

- *At max node:*
  - *receive* α *and* β *values from parent*
  - *expand children one by one*
  - *update* α *as we go*
  - *if* α *ever gets higher than* β, *stop*
  - *won't ever reach this node (return* α)

Alpha-beta pruning

- *At min node:*
  - *receive* α *and* β *values from parent*
  - *expand children one by one*
  - *update* β *as we go*
  - *if* β *ever gets lower than* α, *stop*
  - *won't ever reach this node (return* β*)*

Example



How much do we save?

- *Original tree: b^d nodes*
  - *b = branching factor*
  - *d = depth*
- *If we expand children in random order, pruning will touch $b^{3d/4}$ nodes*
- *Lower bound (best node first): $b^{d/2}$*
- *Can often get close to lower bound w/* ***move ordering heuristics***