15-780: Graduate AI

Lecture 1. Intro & Logic

Geoff Gordon (this lecture)
Tuomas Sandholm
TAs Byron Boots, Sam Ganzfried

Admin

15-780 Graduate Artificial Intelligence Spring 2009

Geoff Gordon and Tuomas Sandholm

School of Computer Science, Carnegie Mellon University

Class Lectures: Tuesdays and Thursdays 10:30-11:50am in 4623 Wean Hall

Recitations: TBA

This course is targeted at graduate students who need to learn about and perform current-day research in artificial intelligence---the discipline of designing intelligent decision-making machines. Techniques from probability, statistics, game theory, algorithms, operations research and optimal control are increasingly important tools for improving the intelligence and autonomy of machines, whether those machines are robots surveying Antarctica, schedulers moving billions of dollars of inventory, spacecraft deciding which experiments to perform, or vehicles negotiating for lanes on the freeway. This AI course is a review of a selected set of these tools. The course will cover the ideas underlying these tools, their implementation, and how to use them or extend them in your research. Students entering the class should have a pre-existing working knowledge of

http://www.cs.cmu.edu/~ggordon/780/
http://www.cs.cmu.edu/~sandholm/cs15-780S09/

Website highlights

- Book: Russell and Norvig. *Artificial Intelligence: A Modern Approach*, 2nd ed.
- Grading: 4–5 HWs, “mid”term, project
- Project: proposal, 2 interim reports, final report, poster
- Office hours

Website highlights

- *Authoritative source for readings, HWs*
- *Please check the website regularly for readings (for Lec. 1–3, Russell & Norvig Chapters 7–9)*

Background

- *No prerequisites*
- *But, suggest familiarity with at least some of the following:*
  - *Linear algebra*
  - *Calculus*
  - *Algorithms & data structures*
  - *Complexity theory*

Waitlist, Audits

- *If you need us to approve something, send us email*

Course email list

- *15780students@…*
- *domain cs.cmu.edu*
- *To subscribe/unsubscribe:*
  - *email 15780students-request@…*
  - *word "help" in subject or body*

Matlab

- *Should all have access to Matlab via school computers*
  - *Those with access to CS license servers, please use if possible*
  - *Limited number of Andrew licenses*
- *Tutorial TBA soon*
- *HWs: please use C, C++, Java, or Matlab*

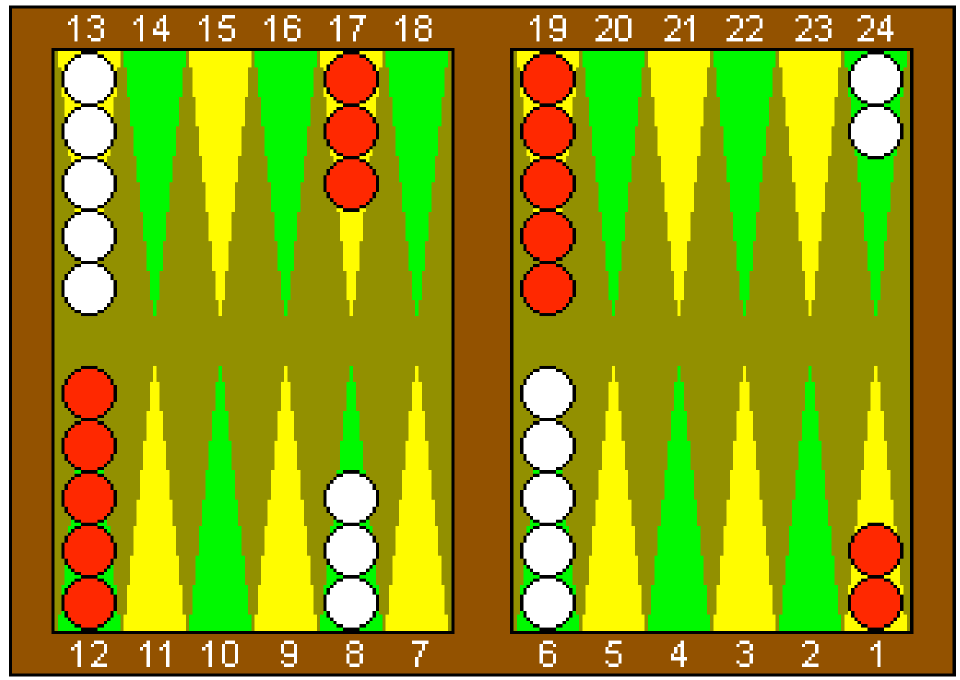
Intro

What is AI?

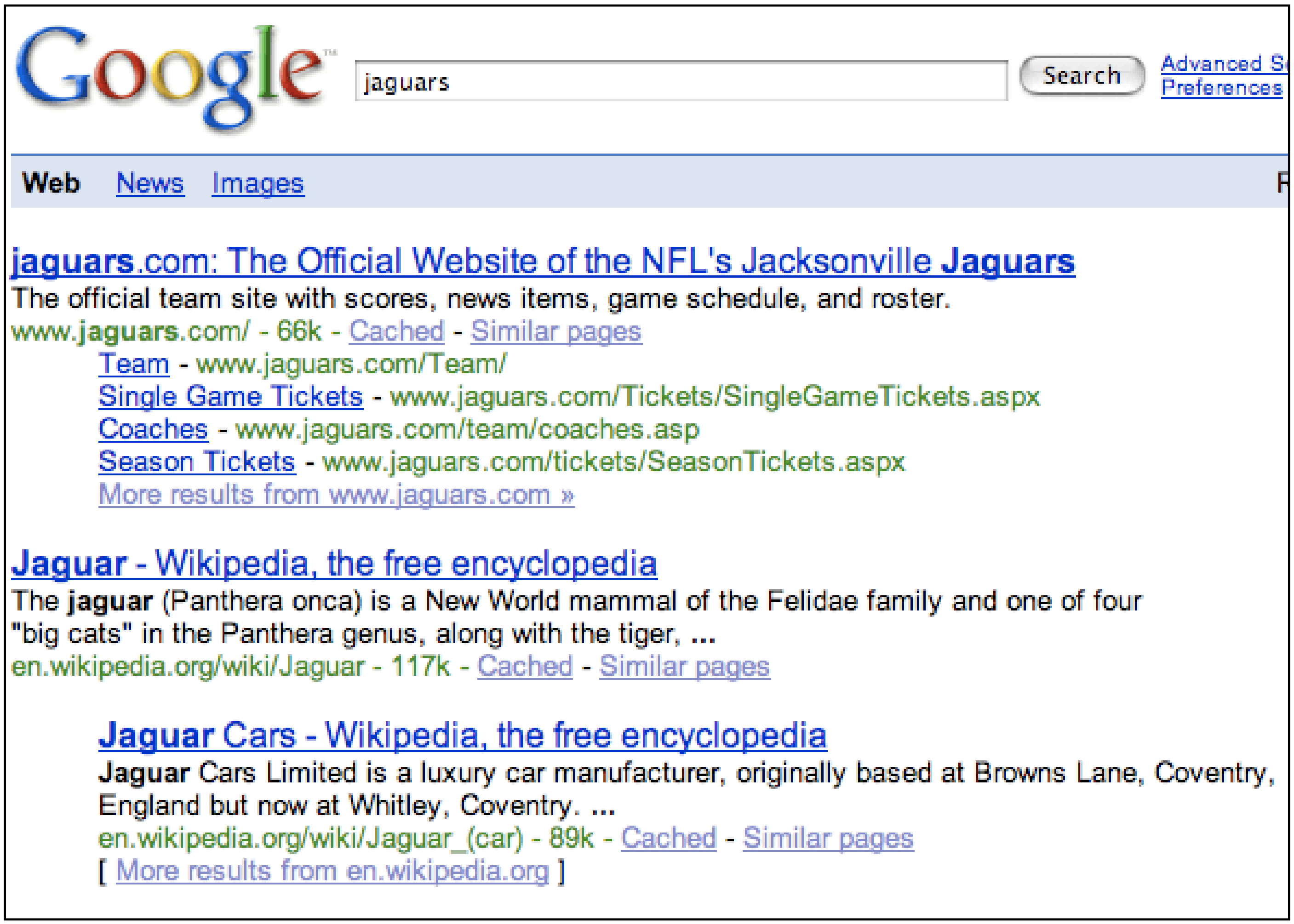
- *Easy part: A*
- *Hard part: I*
  - *Anything we don't know how to make a computer do yet*
  - *Corollary: once we do it, it isn't AI anymore :-)*

Definition by examples

- *Card games*
  - *Poker*
  - *Bridge*
- *Board games*
  - *Deep Blue*
  - *TD-Gammon*
  - *Samuels's checkers player*

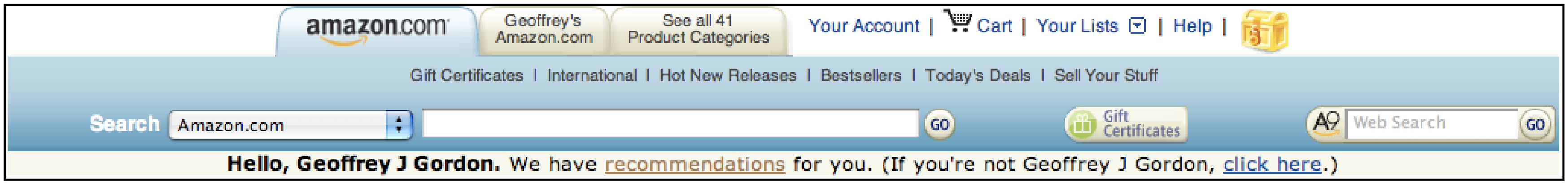


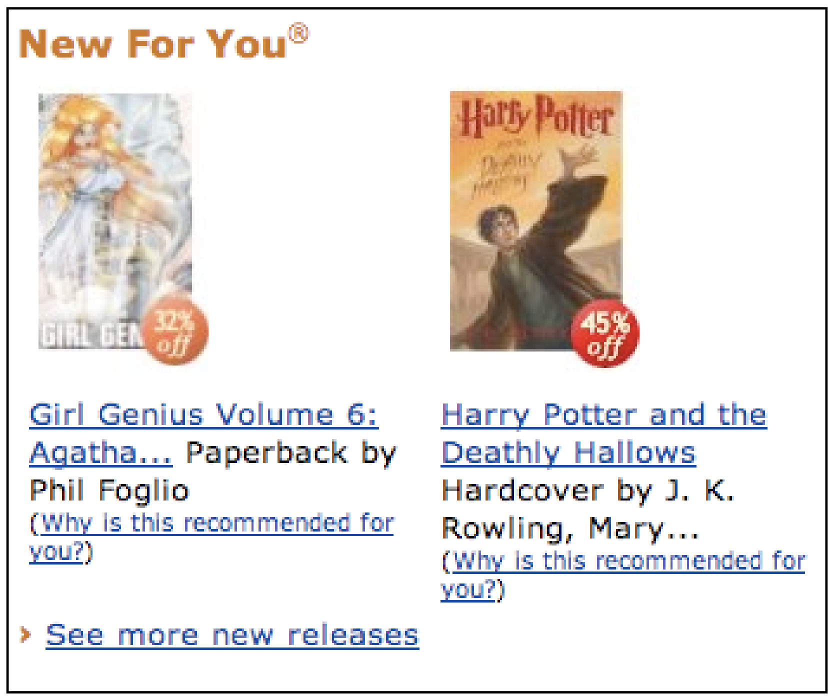
Web search


Google
jaguars
Search
Advanced S
Preferences
Web
News
Images
jaguars.com: The Official Website of the NFL's Jacksonville Jaguars
The official team site with scores, news items, game schedule, and roster.
www.jaguars.com/ - 66k - Cached - Similar pages
Team - www.jaguars.com/Team/
Single Game Tickets - www.jaguars.com/Tickets/SingleGameTickets.aspx
Coaches - www.jaguars.com/team/coaches.asp
Season Tickets - www.jaguars.com/tickets/SeasonTickets.aspx
More results from www.jaguars.com »
Jaguar - Wikipedia, the free encyclopedia
The jaguar (Panthera onca) is a New World mammal of the Felidae family and one of four "big cats" in the Panthera genus, along with the tiger, ...
en.wikipedia.org/wiki/Jaguar - 117k - Cached - Similar pages
Jaguar Cars - Wikipedia, the free encyclopedia
Jaguar Cars Limited is a luxury car manufacturer, originally based at Browns Lane, Coventry, England but now at Whitley, Coventry. ...
en.wikipedia.org/wiki/Jaguar_(car) - 89k - Cached - Similar pages
[More results from en.wikipedia.org]

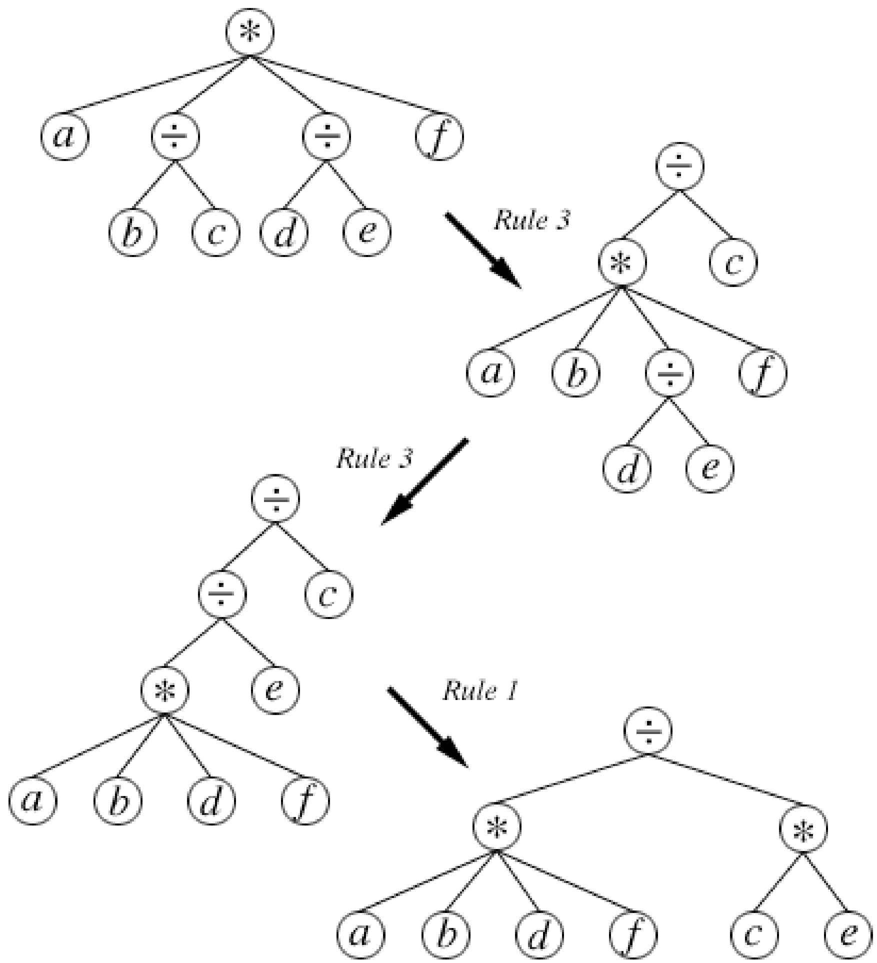
Web search, cont'd

Recommender systems





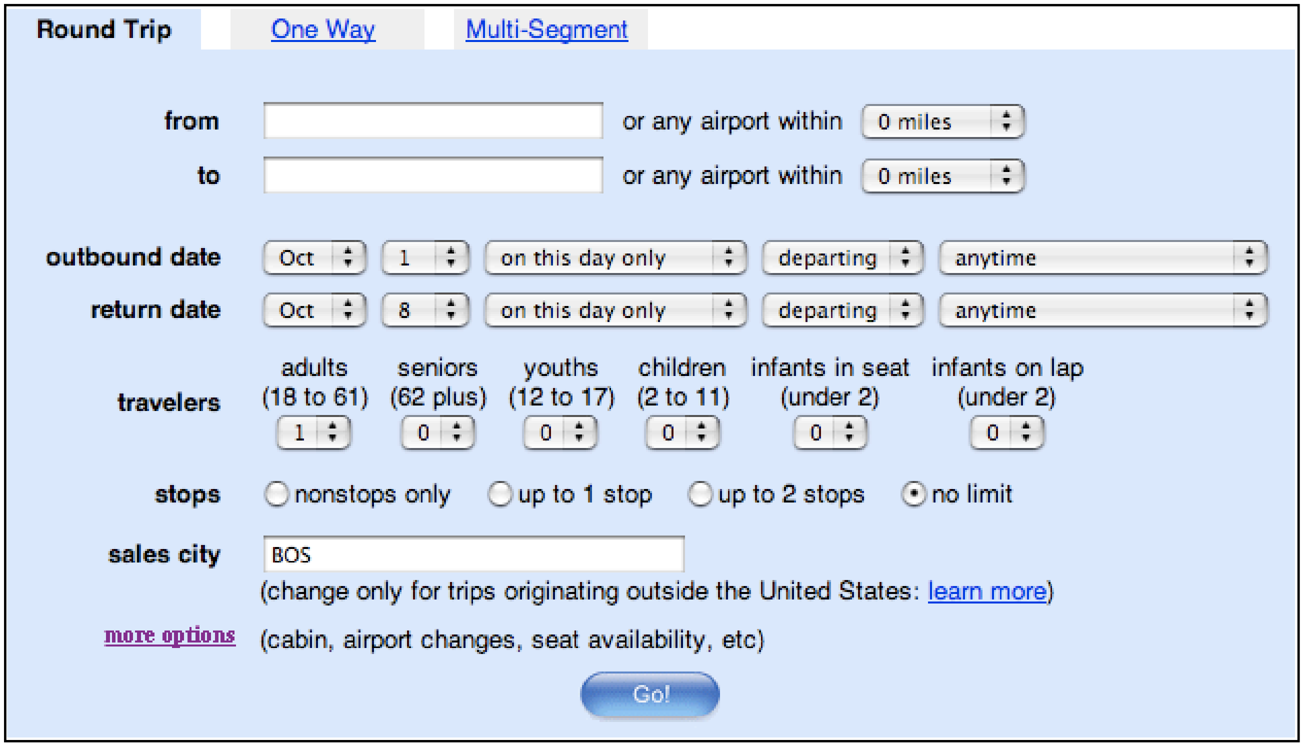
Computer algebra systems



from http://www.math.wpi.edu/IQP/BVCalcHist/calctoc.html

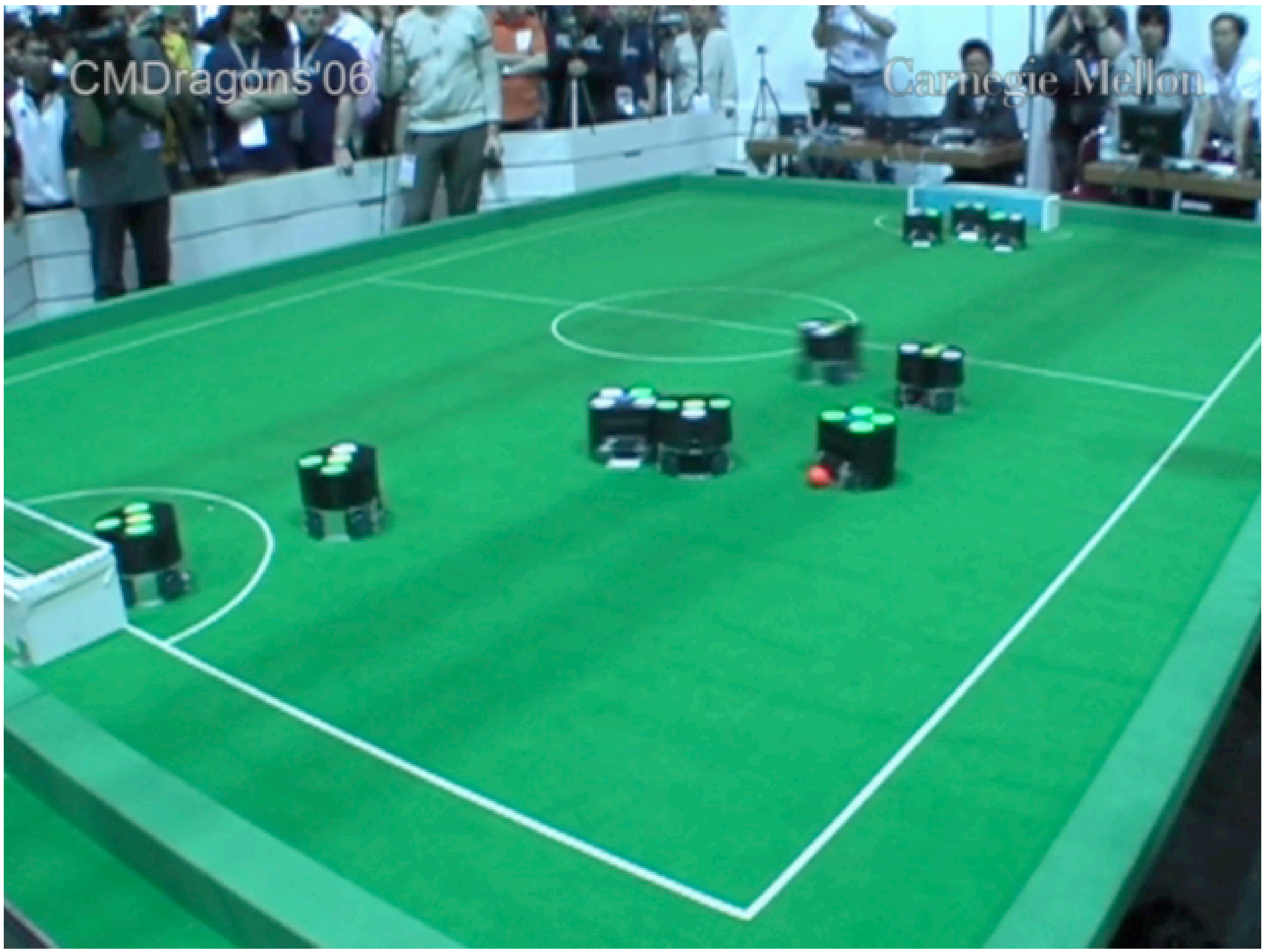
Grand Challenge road race

Getting from A to B



- *ITA software (http://beta.itasoftware.com)*

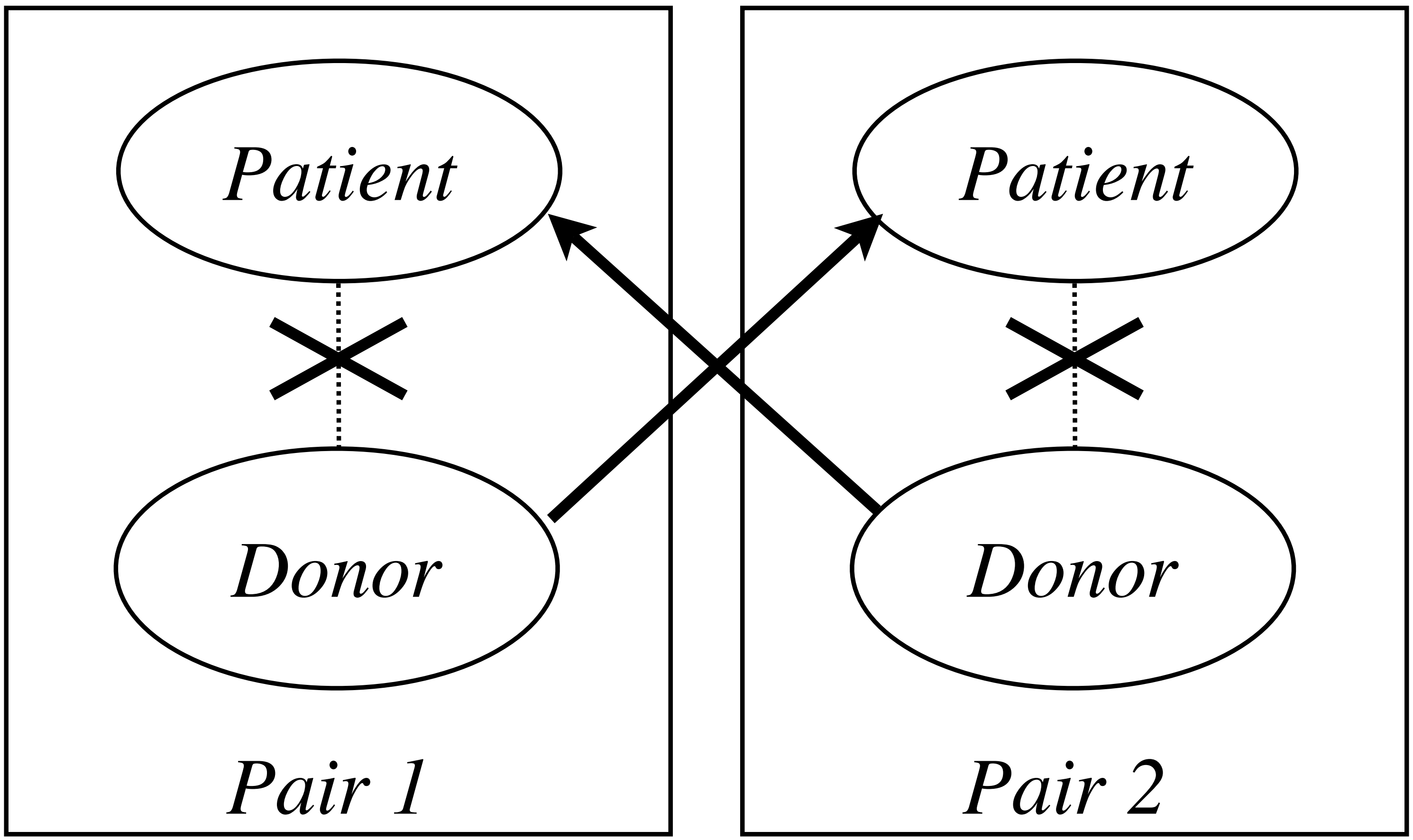
Robocup



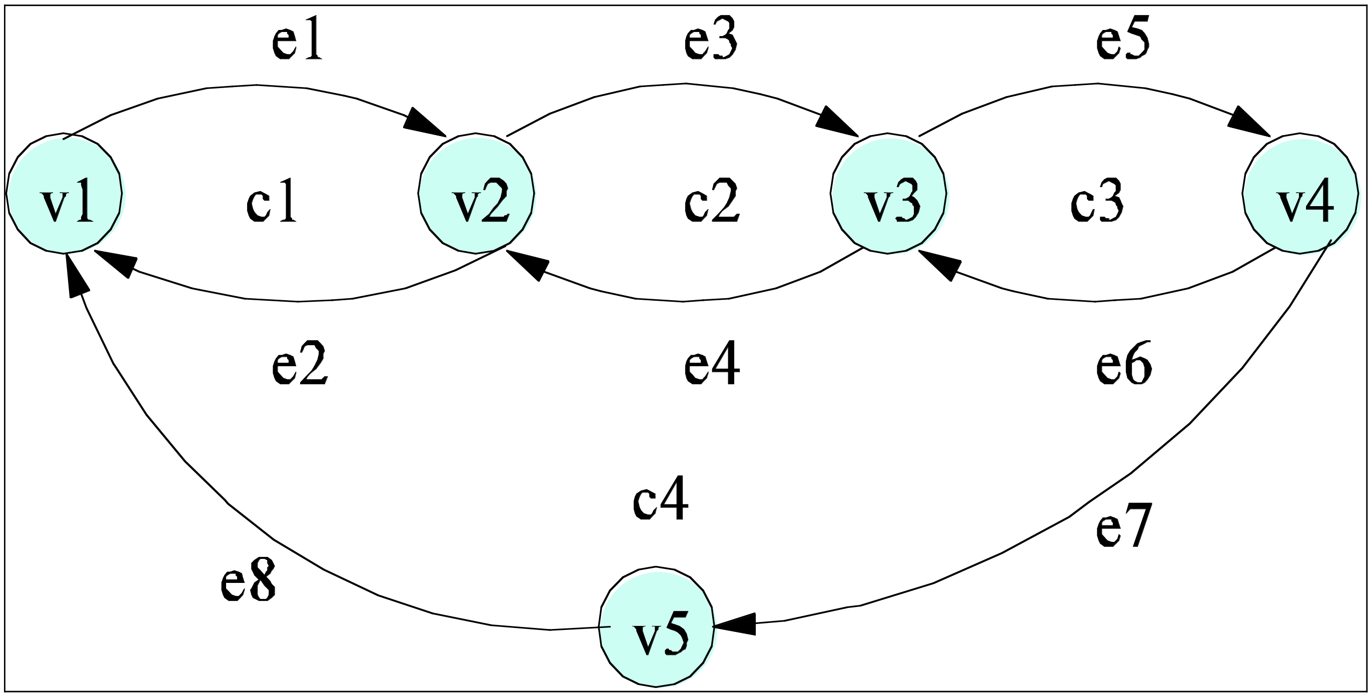
Kidney exchange

- *In US, ≥ 50,000/yr get lethal kidney disease*
- *Cure = transplant, but donor must be compatible (blood type, tissue type, etc.)*
- *Wait list for cadaver kidneys: 2–5 years*
- *Live donors: have 2 kidneys, can survive w/ 1*
- *Illegal to buy/sell, but altruists/friends/family donate*

Kidney Exchange



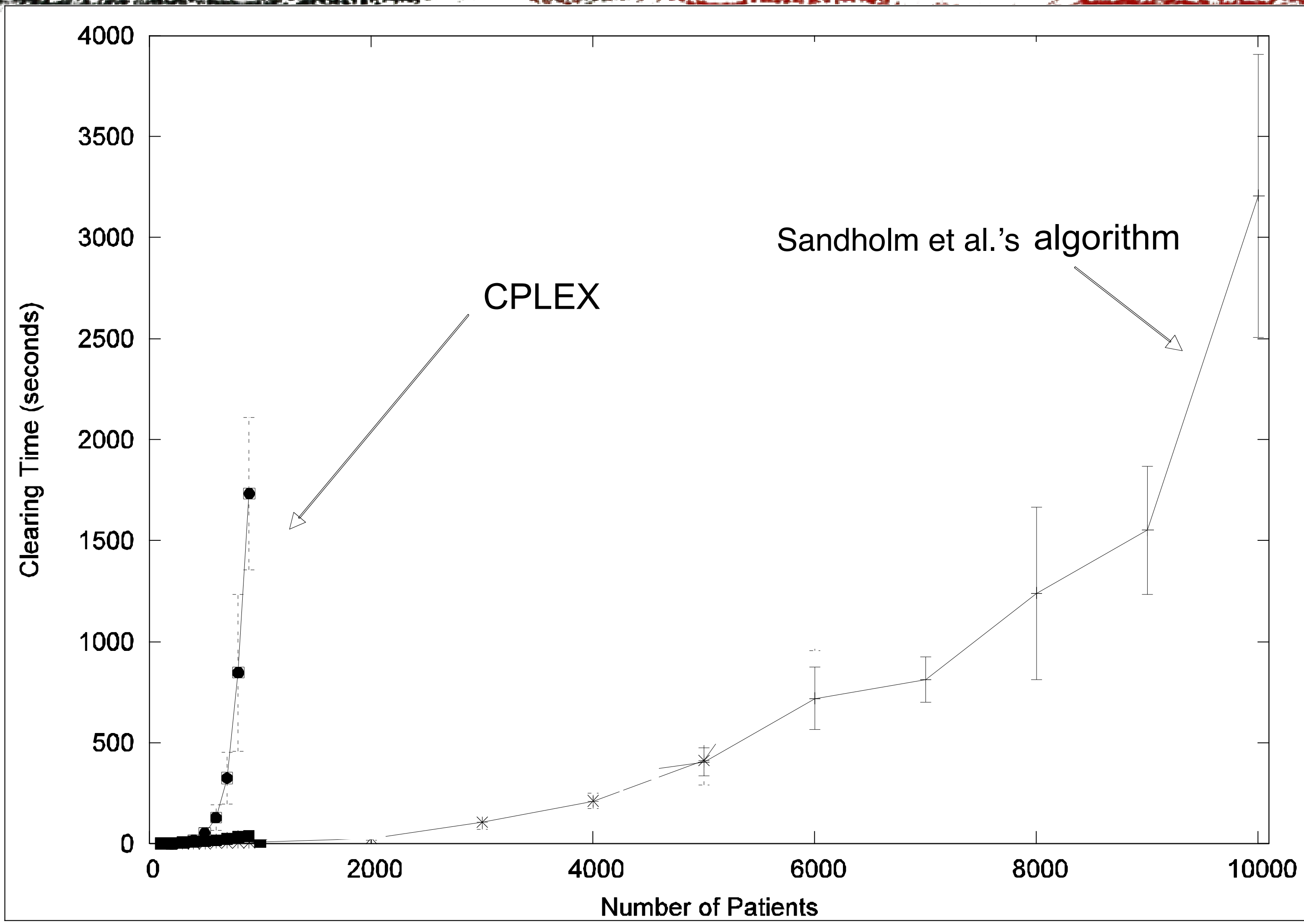
Optimization: cycle cover



Cycle length constraint => extremely hard (NP-complete) combinatorial optimization problem

National market predicted to have 10,000 patients at any one time

Optimization performance

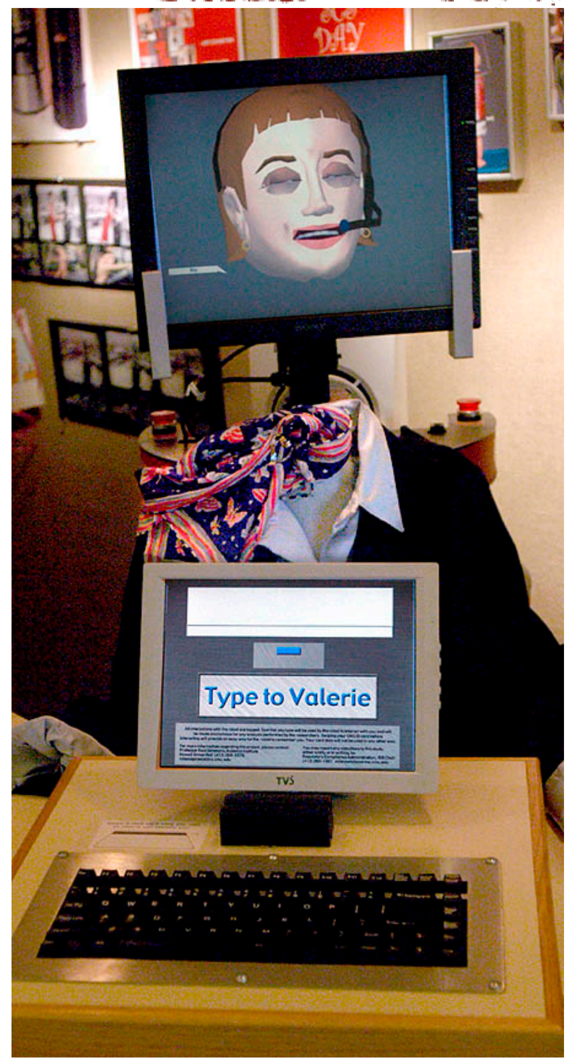


More examples

- *Motor skills: riding a bicycle, learning to walk, playing pool, …*
- *Vision*

More examples

- *Valerie and Tank, the Roboceptionists*
- *Social skills: attending a party, giving directions, …*



More examples

- *Natural language*
- *Speech recognition*

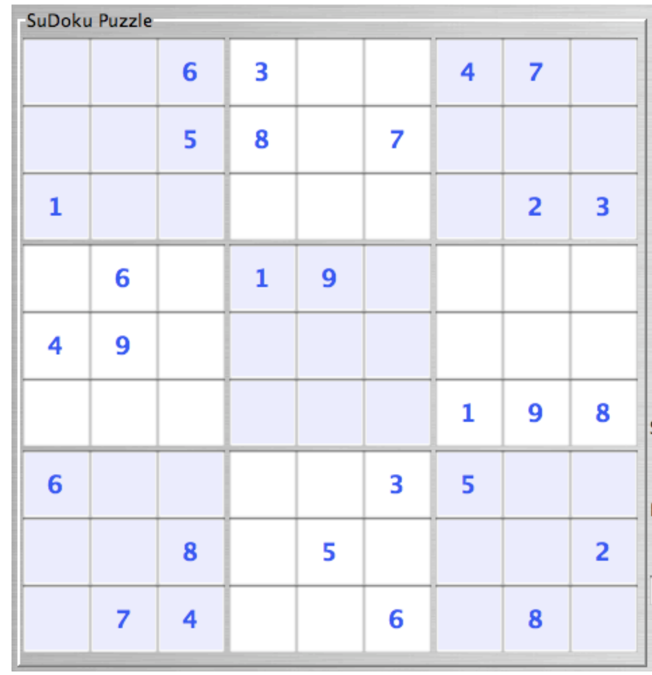
Common threads

- *Finding the needle in the haystack*
  - *Search*
  - *Optimization*
  - *Summation / integration*
- *Set the problem up well (so that we can apply a standard algorithm)*

Common threads

- *Managing uncertainty*
  - *chance outcomes (e.g., dice)*
  - *sensor uncertainty ("hidden state")*
  - *opponents*
- *The more different types of uncertainty, the harder the problem (and the slower the solution)*

Classic AI



- *No uncertainty, pure search*
  - *Mathematica*
  - *deterministic planning*
  - *Sudoku*
- *This is the topic of Part I of the course*

http://www.cs.ualberta.ca/~aixplore/search/IDA/Applet/SearchApplet.html

Outcome uncertainty

- *In backgammon, don't know ahead of time what the dice will show*
- *When driving down a corridor, wheel slippage causes unexpected deviations*
- *Open a door, find out what's behind it*
- *MDPs (later)*

Sensor uncertainty

- *For given set of immediate measurements, multiple world states may be possible*
- *Image of a handwritten digit → 0, 1, …, 9*
- *Image of room → person locations, identities*
- *Laser rangefinder scan of a corridor → map, robot location*

Sensor uncertainty example

Opponents cause uncertainty

- *In chess, must guess what opponent will do; cannot directly control him/her*
- *Alternating moves (game trees)*
  - *not really uncertain (Part I)*
- *Simultaneous or hidden moves: game theory (later)*

Other agents cause uncertainty

- *In many AI problems, there are other agents who aren't (necessarily) opponents*
  - *Ignore them & pretend part of Nature*
  - *Assume they're opponents (pessimistic)*
  - *Learn to cope with what they do*
  - *Try to convince them to cooperate (paradoxically, this is the hardest case)*
- *More later*

Logic

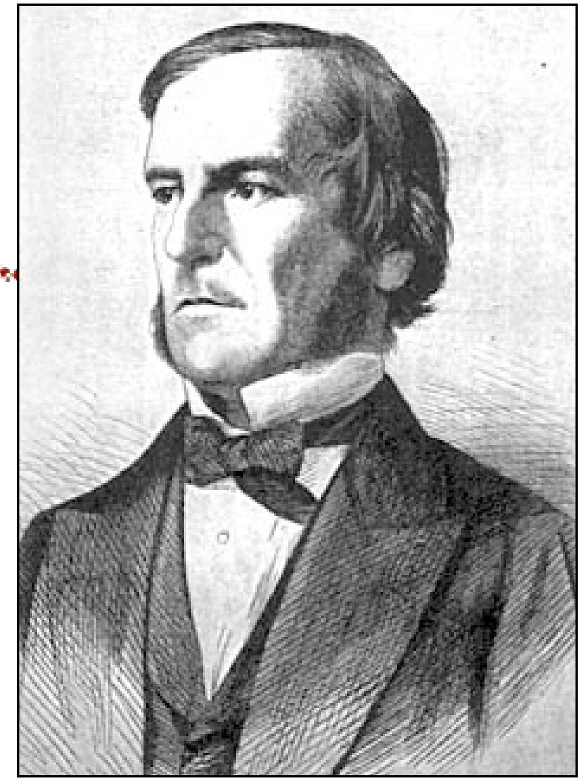
Why logic?

- *Search: for problems like Sudoku, can write compact description of rules*
- *Reasoning: figure out consequences of the knowledge we've given our agent*
- *…and, logical inference is a special case of probabilistic inference*

Propositional logic

- *Constants: T or F*
- *Variables: x, y (values T or F)*
- *Connectives:* $\wedge$, $\vee$, $\neg$
  - *Can get by w/ just NAND*
  - *Sometimes also add others:*

    $\oplus$, $\Rightarrow$, $\Leftrightarrow$, …



George Boole
1815–1864

Propositional logic

- *Build up expressions like* $\neg x \Rightarrow y$
- *Precedence:* $\neg$, $\wedge$, $\vee$, $\Rightarrow$
- *Terminology: variable or constant with or w/o negation =* ***literal***
- *Whole thing =* ***formula*** *or* ***sentence***

Expressive variable names

- *Rather than variable names like x, y, may use names like "rains" or "happy(John)"*
- *For now, "happy(John)" is just a string with no internal structure*
  - *there is no "John"*
  - *$happy(John) \Rightarrow \neg happy(Jack)$ means the same as $x \Rightarrow \neg y$*

But what does it mean?

- *A formula defines a mapping*

  $(\textit{assignment to variables}) \mapsto \{T, F\}$

- *Assignment to variables =* ***model***
- *For example, formula* $\neg x$ *yields mapping:*

x	$\neg x$
T	F
F	T

Questions about models and sentences

- *How many models make a sentence true?*
  - *A sentence is **satisfiable** if it is True in some model*
  - *If not satisfiable, it is a **contradiction** (False in every model)*
  - *A sentence is **valid** if it is True in every model (called a **tautology**)*

Questions about models and sentences

- *How is the variable X set in {some, all} true models?*
- *This is the most frequent question an agent would ask: given my assumptions, can I conclude X? Can I rule X out?*

More truth tables

x	y	$x \wedge y$
T	T	T
T	F	F
F	T	F
F	F	F

x	y	$x \vee y$
T	T	T
T	F	T
F	T	T
F	F	F

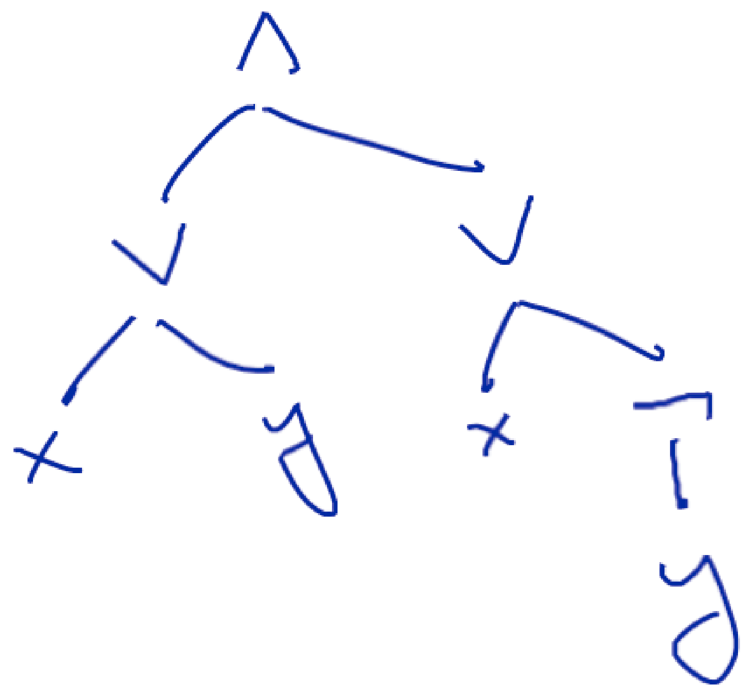
Truth table for implication

- $(a \Rightarrow b)$ *is logically equivalent to* $(\neg a \vee b)$
- *If a is True, b must be True too*
- *If a False, no requirement on b*
- *E.g., "if I go to the movie I will have popcorn": if no movie, may or may not have popcorn*

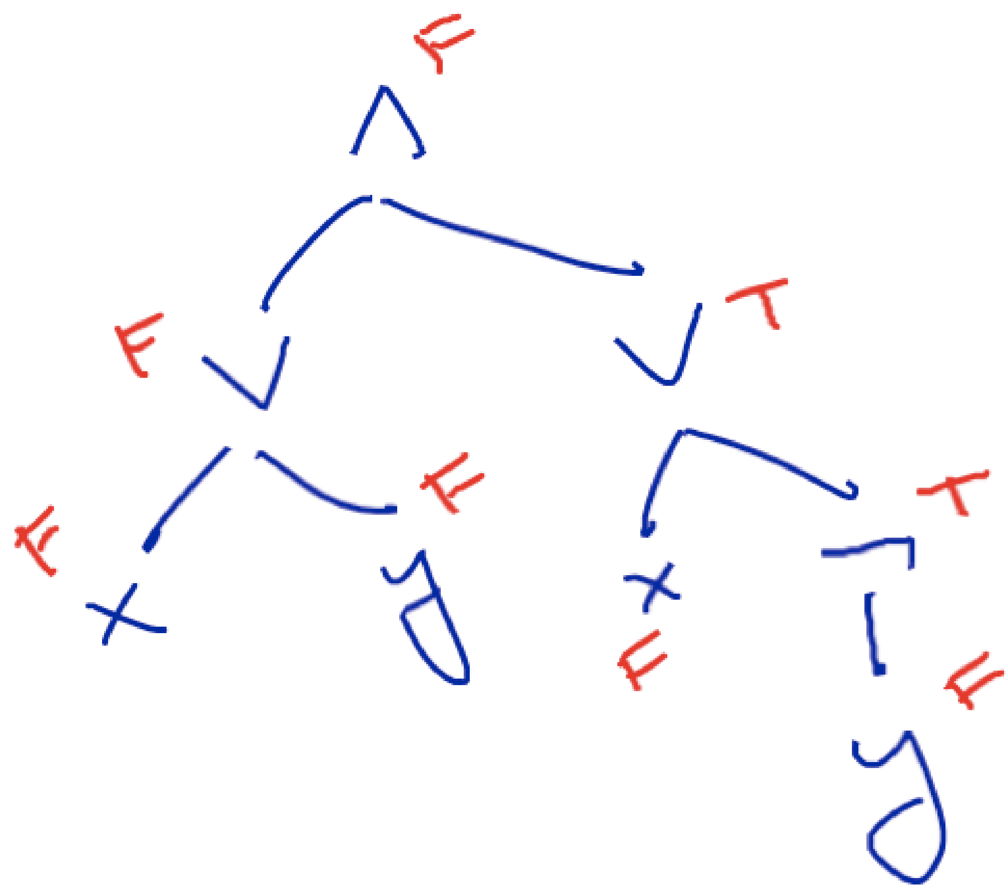
a	b	$a \Rightarrow b$
T	T	T
T	F	F
F	T	T
F	F	T

Complex formulas

- *To evaluate a bigger formula*
  - $(x \lor y) \land (x \lor \neg y)$ *when* $x = F, y = F$
- *Build a parse tree*
- *Fill in variables at leaves using model*
- *Work upwards using truth tables for connectives*

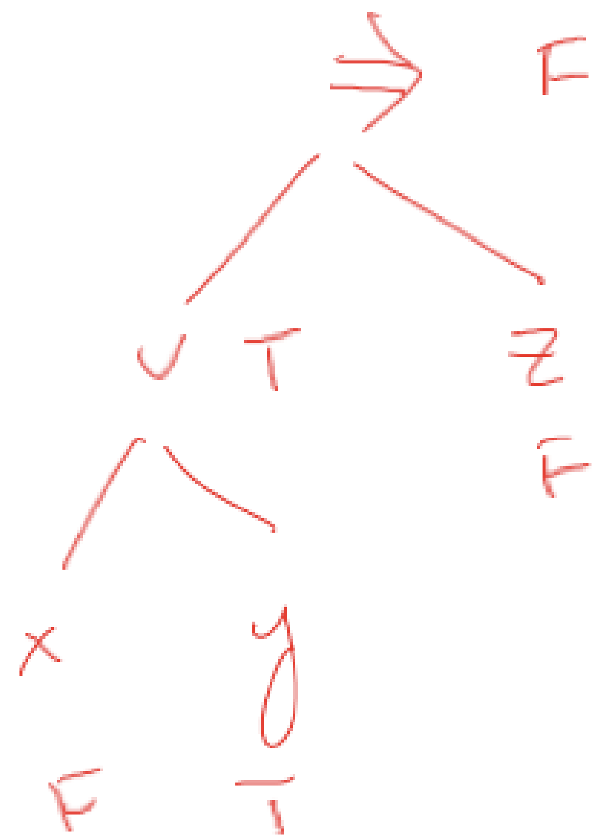


Example



- $(x \vee y) \wedge (x \vee \neg y)$ *when* $x = F, y = F$

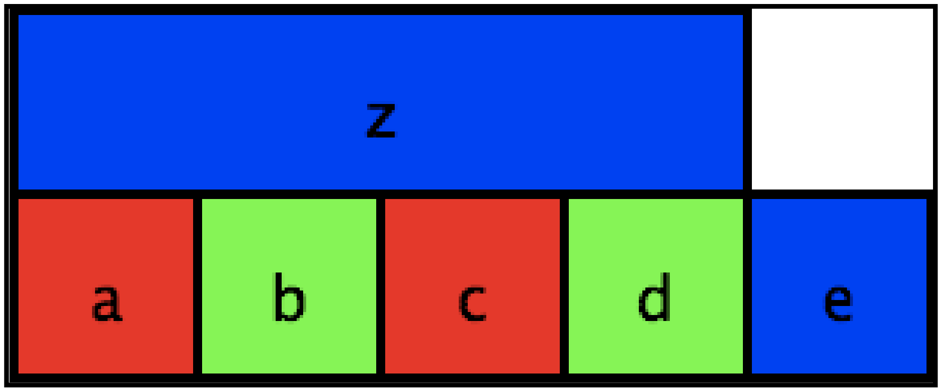
Another example

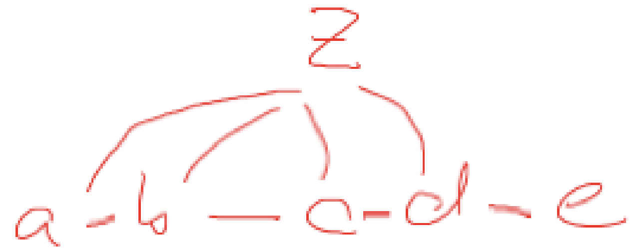


$(x \vee y) \Rightarrow z$ x = F, y = T, z = F

Example

3-coloring





vars: $aR \quad aB \quad aG$

$bR \quad bB \quad \ldots$

constrs: $(aR \vee aB \vee aG) \wedge$

$(bR \vee bB \vee bG) \wedge \ldots$

$\wedge (\overline{aR} \vee \overline{bR}) \wedge (\overline{aB} \vee \overline{bB}) \wedge (\overline{aG} \vee \overline{bG})$

$\wedge (\overline{aR} \vee \overline{zR}) \ldots$

Sudoku

SuDoku Puzzle

		6	3			4	7	
		5	8		7			
1							2	3
	6		1	9				
4	9							
						1	9	8
6					3	5		
		8		5				2
	7	4			6		8	

http://www.cs.qub.ac.uk/~I.Spence/SuDoku/SuDoku.html

Minesweeper

0	0	1	v1		
0	0	1	v2		
0	0	1	v3		
1	1	2	v4		
v8	v7	v6	v5		

V = { v1 , v2 , v3 , v4 , v5 , v6 , v7 , v8 }, D = { B (bomb) , S (space) }

C = { (v1,v2) : { (B , S) , (S,B) } ,(v1,v2,v3) : { (B,S,S) , (S,B,S) , (S,S,B)},...}

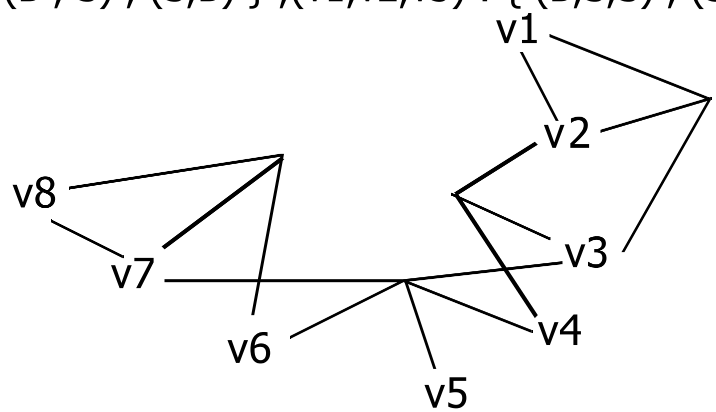


image courtesy Andrew Moore

Propositional planning

init: have(cake)

goal: have(cake), eaten(cake)

eat(cake):

  pre: have(cake)

  eff: -have(cake), eaten(cake)

bake(cake):

  pre: -have(cake)

  eff: have(cake)

Other important logic problems

- *Scheduling (e.g., of factory production)*
- *Facility location*
- *Circuit layout*
- *Multi-robot planning*

- *Important issue: handling uncertainty*

Working with formulas

Truth tables get big fast

x	y	z	$(x \vee y) \Rightarrow z$
T	T	T	
T	T	F	
T	F	T	
T	F	F	
F	T	T	
F	T	F	
F	F	T	
F	F	F	

Truth tables get big fast

x	y	z	a	$(x \lor y \lor a) \Rightarrow z$
T	T	T	T	
T	T	F	T	
T	F	T	T	
T	F	F	T	
F	T	T	T	
F	T	F	T	
F	F	T	T	
F	F	F	T	
T	T	T	F	
T	T	F	F	
T	F	T	F	
T	F	F	F	
F	T	T	F	
F	T	F	F	
F	F	T	F	
F	F	F	F	

Definitions

- *Two sentences are* ***equivalent****, A ≡ B, if they have same truth value in every model*
  - *(rains ⇒ pours) ≡ (¬rains ∨ pours)*
  - *reflexive, transitive, commutative*
- ***Simplifying*** *= transforming a formula into a shorter*, equivalent formula*

Transformation rules

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha & \text{double-negation elimination}
\end{aligned}
$$

$$
\begin{aligned}
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

α, β, γ are arbitrary formulas

More rules

$$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) \quad \text{implication elimination}$$

$$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination}$$

$$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) \quad \text{de Morgan}$$

$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \quad \text{de Morgan}$$

α, β are arbitrary formulas

Still more rules…

- *…can be derived from truth tables*
- *For example:*
  - $(a \vee \neg a) \equiv True$
  - $(True \vee a) \equiv True$
  - $(False \wedge a) \equiv False$

Example

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

distrib.

de Morgan

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

$((\neg b \wedge \neg c) \vee a) \wedge \ldots$ commute.

$(\neg b \wedge \neg c) \vee \underline{(a \wedge \neg a)}$ distrib.

$(\neg b \wedge \neg c) \vee F$

$(\neg b \wedge \neg c)$

Normal Forms

Normal forms

- *A normal form is a standard way of writing a formula*
- *E.g.,* ***conjunctive normal form*** *(CNF)*
  - *conjunction of disjunctions of literals*
  - $(x \lor y \lor \neg z) \land (x \lor \neg y) \land (z)$
  - *Each disjunct called a* ***clause***
- *Any formula can be transformed into CNF w/o changing meaning*

CNF cont'd

$$\begin{array}{|l|}\hline \mathit{happy}(\mathit{John}) \wedge \\ (\neg \mathit{happy}(\mathit{Bill}) \vee \mathit{happy}(\mathit{Sue})) \wedge \\ \mathit{man}(\mathit{Socrates}) \wedge \\ (\neg \mathit{man}(\mathit{Socrates}) \vee \mathit{mortal}(\mathit{Socrates})) \\ \hline \end{array}$$

- *Often used for storage of knowledge database*
  - *called **knowledge base** or KB*
- *Can add new clauses as we find them out*
- *Each clause in KB is separately true (if KB is)*

Another normal form: DNF

- *DNF = disjunctive normal form = disjunction of conjunctions of literals*
- *Doesn't compose the way CNF does: can't just add new conjuncts w/o changing meaning of KB*
- *Example:*

$$\boxed{\begin{array}{c} (rains \lor \neg pours) \land fishing \\ \equiv \\ (rains \land fishing) \lor (\neg pours \land fishing) \end{array}}$$

to CNF

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

$\neg d \wedge \neg(e \wedge f)$

$(\neg e \vee \neg f)$

to DNF

$$(a \vee b \vee \neg c) \wedge \underline{\neg(d \vee (e \wedge f))} \wedge \underline{(c \vee d \vee e)}$$

$$\neg(d \vee (e \wedge f)) \wedge c) \vee (\neg(d \vee (e \wedge f)) \wedge d)$$
$$\vee (\neg(d \vee (e \wedge f)) \wedge e)$$

$$\underbrace{\alpha}_{\text{big}} \wedge (\underbrace{\beta \vee \gamma}_{\text{small}}) \longrightarrow \underbrace{\alpha \wedge \beta \vee \alpha \wedge \gamma}_{\substack{\text{almost } 2x \\ \text{big}}}$$

Transforming to CNF or DNF

- *Naive algorithm:*
  - *replace all connectives with* $\wedge \vee \neg$
  - *move negations inward using De Morgan's laws and double-negation*
  - *repeatedly distribute over* $\wedge$ *over* $\vee$ *for DNF (*$\vee$ *over* $\wedge$ *for CNF)*

Example

- *Put the following formula in CNF*

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

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

Example

- *Now try DNF*

$$(a \lor b \lor \neg c) \land \neg(d \lor (e \land f)) \land (c \lor d \lor e)$$

Discussion

- *Problem with naive algorithm: it's exponential! (Space, time, size of result.)*
- *Each use of distributivity can almost double the size of a subformula*

A smarter transformation

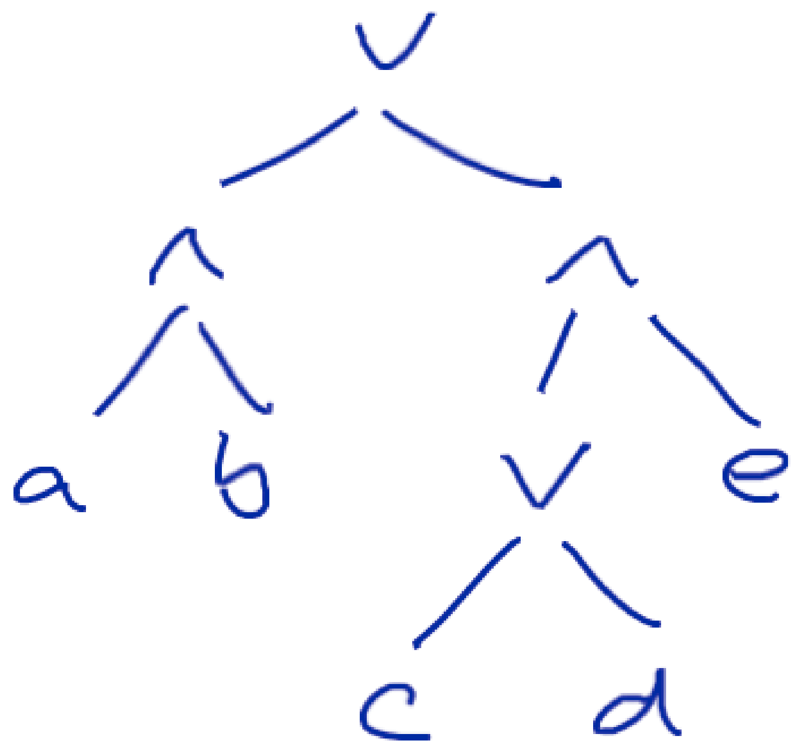
- *Can we avoid exponential blowup in CNF?*
- *Yes, if we're willing to introduce new variables*
- *G. Tseitin. On the complexity of derivation in propositional calculus. Studies in Constrained Mathematics and Mathematical Logic, 1968.*

Tseitin example

- *Put the following formula in CNF:*

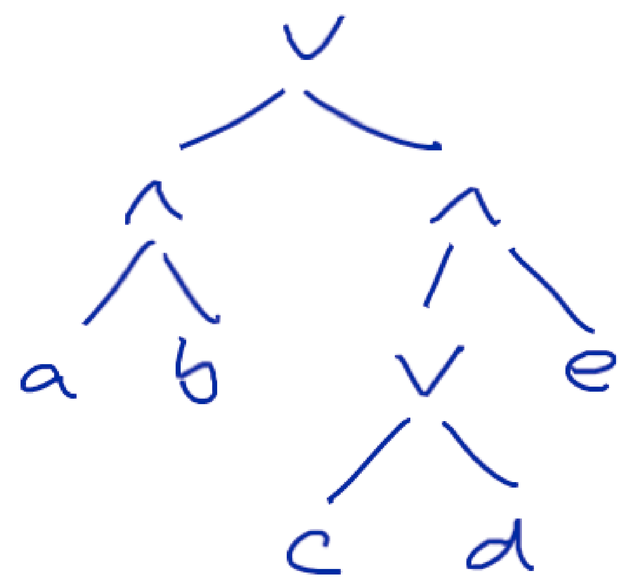
  $(a \wedge b) \vee ((c \vee d) \wedge e)$

- *Parse tree:*



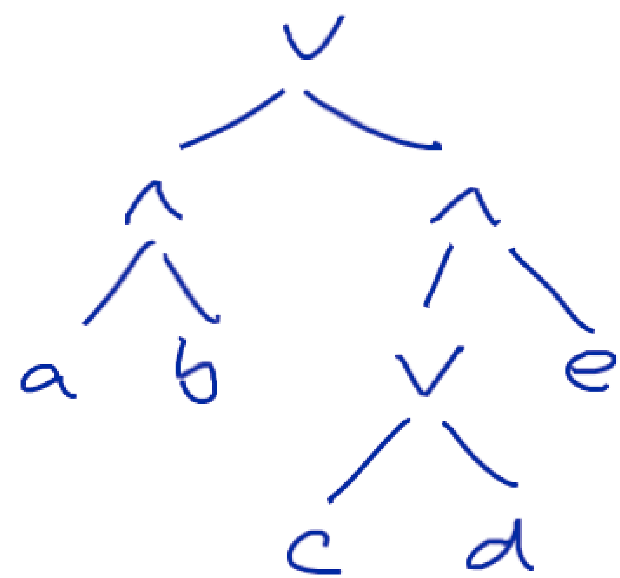
Tseitin transformation

- *Introduce temporary variables*
  - $x = (a \wedge b)$
  - $y = (c \vee d)$
  - $z = (y \wedge e)$



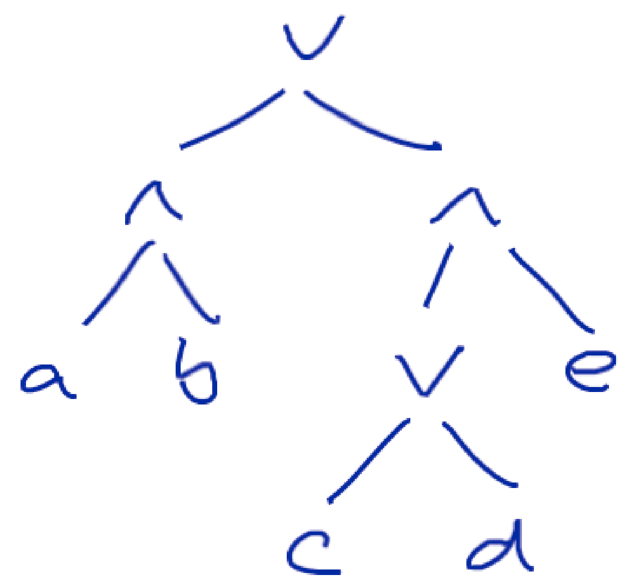
Tseitin transformation

- *To ensure* $x = (a \wedge b)$, *want*
  - $x \Rightarrow (a \wedge b)$
  - $(a \wedge b) \Rightarrow x$



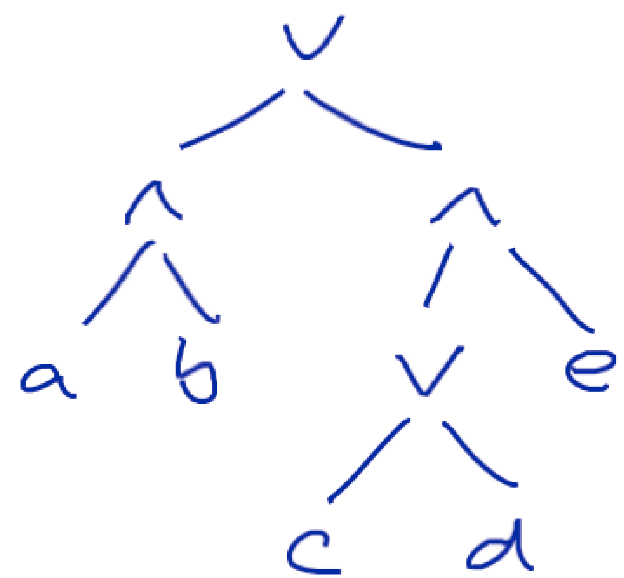
Tseitin transformation

- $x \Rightarrow (a \wedge b)$
- $(\neg x \vee (a \wedge b))$
- $(\neg x \vee a) \wedge (\neg x \vee b)$



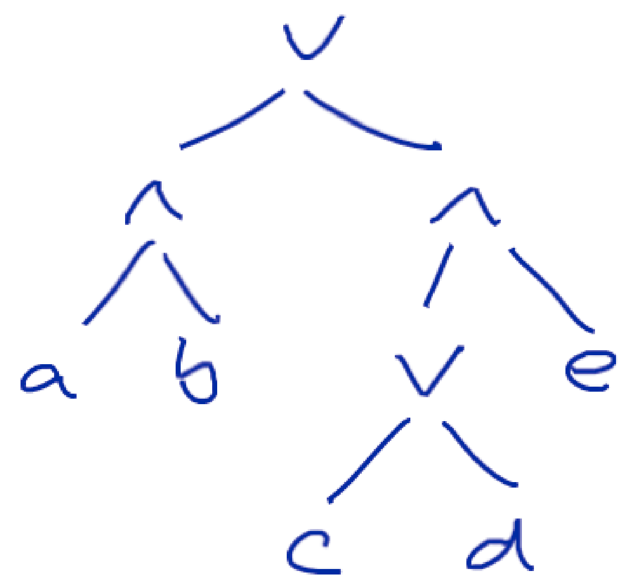
Tseitin transformation

- $(a \wedge b) \Rightarrow x$
- $(\neg(a \wedge b) \vee x)$
- $(\neg a \vee \neg b \vee x)$



Tseitin transformation

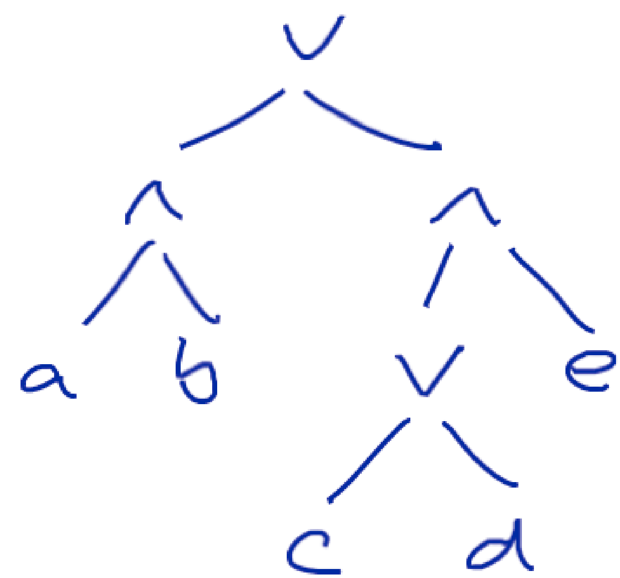
- *To ensure* $y = (c \vee d)$*, want*
  - $y \Rightarrow (c \vee d)$
  - $(c \vee d) \Rightarrow y$



Tseitin transformation

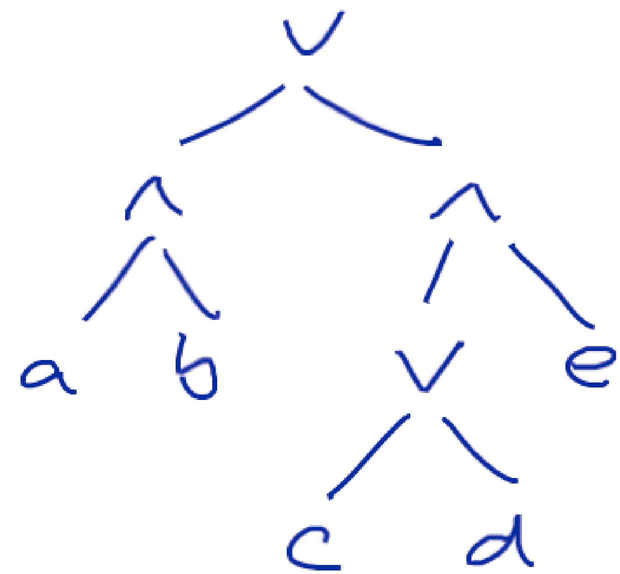
- $y \Rightarrow (c \vee d)$
- $(\neg y \vee c \vee d)$

- $(c \vee d) \Rightarrow y$
- $((\neg c \wedge \neg d) \vee y)$
- $(\neg c \vee y) \wedge (\neg d \vee y)$

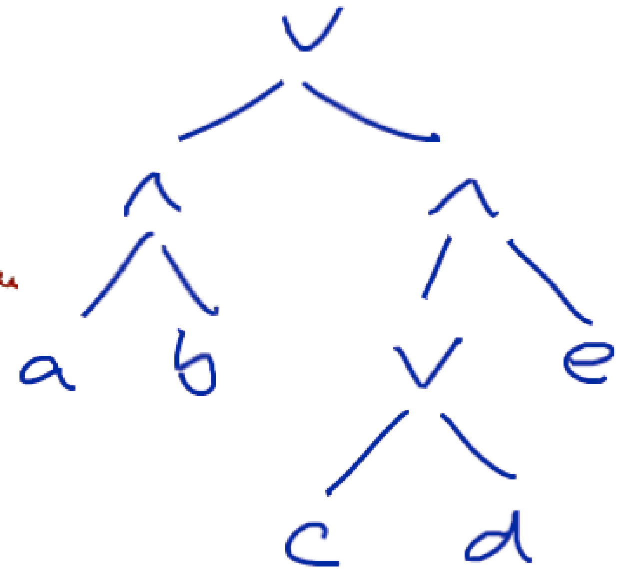


Tseitin transformation

- *Finally*, $z = (y \wedge e)$
- $z \Rightarrow (y \wedge e) \equiv (\neg z \vee y) \wedge (\neg z \vee e)$
- $(y \wedge e) \Rightarrow z \equiv (\neg y \vee \neg e \vee z)$



Tseitin end result



$$(a \wedge b) \vee ((c \vee d) \wedge e) \equiv$$

$$(\neg x \vee a) \wedge (\neg x \vee b) \wedge (\neg a \vee \neg b \vee x) \wedge$$
$$(\neg y \vee c \vee d) \wedge (\neg c \vee y) \wedge (\neg d \vee y) \wedge$$
$$(\neg z \vee y) \wedge (\neg z \vee e) \wedge (\neg y \vee \neg e \vee z) \wedge$$
$$(x \vee z)$$