	M 11-
	Midterm
Lisp	· = 15 only for numbers I is hely all the second as a self of the
	· equal is to check if the contents are the same
	eql is to check if they are actually the same object in memory must all a manage of
-	· AND - If T, then returns value of the final expression
	if F, then refuns NZLaman , then plans angel and 150 220 50
20014	• OR - returns the first expression that is true
leminorpools (on)	· (cond (bexp exp, expn))
	· let - creates local bindings for local variables [variables assigned in parallel]
	· lett - same as let, but assigns variables sequentrally instead
	Walnus en Sources (all findings a odd allow whom all of larget.
Search	· Three components - Instral State, Final State, Actions, (Herristic)
portion it set	Actions have a particular cost, want to satisfy some conditions/constraints
[ (al "ol 2 con	· Actums are deterministic (lactron > I state) or non-deterministic
	· Successor Function - takes a state, tells us possible next states
	based on a legal action should be to be seen to the
(	· Search Process and annual ments, and all all the many and a ser special of the
	(1) Generate Children de de de la selection de la constant de la c
((n)	(2) Expand - Check if you reached the goal yet
	If not at goal yet, generate children
1	Fringe/Frontier - set of children (leaf nodes) at the end of the visited path 92)
	· Evaluation of Search Stratogres
	1) completeness - does it always find a solution if one exists?
	@ time complexity - number of nudes generated declarated
hobas	3 space complexity - max number of nodes in memory to find a solution
	9 Optimality - does it find the least-cost solution?
Jun T	· Branching Factor (b) - mux number of children per pavent
	· Depth (d) - depth of least cost solution
	· Max depth of Search Tree (m) [Alote: this earld be infinite if no solution]
UI	BFS: O(bd) time/space complexity, complete, optimal
UI	DFS: O(bm) time, O(bm) space, not complete, not optimal
UI	Limited DFS: limit depth to L so we don't search infinitely
Bearing In	→ Not complete if solution below depth limit, complete if d ∈ L
	O(b) time, O(bL) space, Not optimal (same as DFS)

UŢ	"Iterative Deepening (ID) Search - Limited DFS, gradually increase depth until solution
	40(bd) time, O(bd) space, complete, optimal
g(n) : actual path UI	
real dist	· E: Smallest cost attained by any action, C": depth of optimal solution
	L> O(bc/e) time/space complexity, complete, optimal
h(n): estimated path I	1) Best-First Sourch (Greety Search, Informed) - UCS but with straight-line distances
La Hecristic	· Straight - line distances from current point to end point, good for estimates (not always optimal)
[ leil ma	
f(n): heurstic I	(2) A* Search: fcn) = g(n) + hcn) - A* is optimal if heuristic is admissible
	· travel to the mode with the smallest f(n)
	Uninformed (UI) search: Only have access to problem definition, no extra information
27 m = 1	Informed (I) Search: have access to problem definition, along with heuristic function
	Admissible Heuristic: estimated cost & cost of optimal path from n to goal [hon) & h (n)]
	· basically - admissible heuristic cost should never overestimate the
	cost of an actual mode and contain a police on he haved
	· if there are two admissible heverstics, choose the one that dominates (greater)
	L> H,=7, Hz=18 Hz>H, -> Chose Hz, It will solve more effectantly
	→ H, and Hz, neither one dominates -> use Ms(n) = max(h,(n), hz(n))
	middle to danguage day the groups had to
CSP	· Given a set of variables, values, and constraints, find a solution to the problem
	· Unary/Binary constraints - one variable involved, two variables involved
	· High-order constraints - more than 2 writables involved
	· Hard constaints - absolutely can not violate these
(85) a(b)	· Soft constraints - not all constraints equally weighted, can violate some if needed
	· SAT - given a statement with different variables and valves, return
	a list of value assignments to the variables so the statement returns True
	CSP as Search modules the long de Mark
part of the	· Initral State - empty variable assignment {}
	· Final State Test
	· Actions
	· State - partral variable assignment
	· Successor Function - generates Next states based on current state and legal actions
	(230 to men) families and 1 selection of 1430 men (2430

CSP Search Tree {}  Tree Depth: N  [DFS here]  X+0  X+1  In of values: d [0,1]  In of declars: d [0,1]  In of values: d [0,1]  In of values: d [0,1]  In of declars: d [0,1]  In of values: d [0,1]  In of declars: d [0,1]  In of values: d [0,1]  In of declars: d [0,	5		
Timprovements of DFS for CSP   Backtrack Search - DFS, but entry generate feasible children   Lo "feasible" - No strate constraint is violated	100	CS P Search Tree Tree Depth: North Tree Depth: N	z britani) (
Improvements of DFS for CSP  (Backtrack Search - DFS, but only generate feasible children  by "feasible" - no single constraint is violated  • Ex: N. Queens Pidden. Place I Queen in a spot, then place 2 <sup>nd</sup> Queen.  If 2 <sup>nd</sup> Queen v invalid, "backtrack" civil place it again in different spot  (Drange ordering)  • Change order of values [s.e. x:1 on left branch, x = 0 on right branch]  • this permites the leaf nodes, can help frad solution auricher  • value ordering keuristic - assign value for the variable with A  the most restraints (variable with fowest legal values) first  (DVariable Ordering)  • change order of variables [1.e. Instead of x > y > z, do x > z > y  • this primes the tree so there's less to search through  • variable ordering herristic - assign value for the variable with  the least restraints (variable with most legal values) first  (DForward Checking)  • Maintain a set of legal values for each variable  • Men you assign a value to one variable, update legal values  of other variables accordingly  • declare "bad state" when some variable has no legal values left  (PAIC Consistency  • Create state diagram for all variables > All arcs chald be consistent  {R,G,B3} {R,B3} {B}		[DFS here] x=0 x=1 # of values: d = [0,1]	
Improvements of DFS for CSP  (1) Backtrack Search - DFS, but only generate feasible children  (2) "feasible" - no single constraint is violated  (Ex: N. Queens Problem - Place I Queen in a spot, then place 2nd Queen.  If 2nd Queen is invalid, "backtrack" and place it again in different spot  (2) Value Ordering  (3) Value Ordering  (4) Value Ordering  (5) Value Ordering  (5) Value Ordering  (6) Value Ordering heuristic - assign value for the variable with the most restraints (variable with fonest logal values) first  (3) Vanable Ordering  (6) Vanable Ordering  (7) Variable ordering  (8) Vanable Ordering  (8) Variable ordering heuristic - assign value for the variable with the least restraints (variable with most legal values) first  (9) Forward Checking  (9) Mannan a set of legal values for each variable  (9) Forward Checking  (1) Mannan a set of legal values for each variable  (1) When you assign a value to one variable, update legal values  of other variables accordingly  (1) declare "bad state" when some variable has no legal values left  (6) Arc Consistency  (1) Create state diagram for all variables in All arcs chould be consistent  (8) R, 6, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8,		4=0 1=1 4:0 7=1 # of leaves: di	
Bocktrack Search - DFS, but only generate feasible children  1> "feasible" - No single constraint is violated  *Ex: No Gueens Problem - Place I Queen in a spot, them place 2nd Queen.  If 2nd Queen is invalid, "backtrack" and place it again in different spot  (Nature Ordering  • Change order of values five x=1 on left branch, x=0 on right branch]  • this permotes the leaf nodes, can help find solution queeker  • value ordering heuristic - assign value for the variable with the most rectraints (variable with fowest logal values) first  (Nanahle Ordering  • change order of variables [i.e. instead of x→ y→ z, do x → z → y]  • this prohes the tree so there's less to search through  • variable ordering heuristic - assign value for the variable with the least restraints (variable with most legal values) first  (Forward Checking  • Maintain a set of legal values for each variable  • When you assign a value for each variable  • When you assign a value for each variable  • When you assign a value for each variable  • When you assign a value for each variable  • When you assign a value for each variable  • When you assign a value for each variable  • When you assign a value for each variable  • Acclare "bad state" when some variable has notegal values left  (Acclare "bad state" when some variable has notegal values left  (Acclare state diagram for all variables → All arcs chould be consistent  {R, G, B3 {R, B3 {B}}			
Backtrack Search - DFS, but only generate feasible children  b "feasible" - no single constraint is violated  • Ex: N. Queens Problem - Place I Queen in a spot, then place 2nd Queen.  If 2nd Queen or invalid, "backtrack" and place it again in different spot  (Nature Ordering  • Change order of values [s.e. x=1 on left branch, x=0 on right branch]  • this permites the teat nodes, can help final solution quicker  • value ordering heuristic - assign value for the variable with the most restraints (variable with fowest logal values) first  (Nanahe Ordering  • change order of variables [1.e. instead of x + y - z, do x + z - y]  • this primes the tree so there's less to search through  • variable ordering heuristic - assign value for the variable with the least restraints (variable with most legal values) first  (Forward Checking  • Maintain a set of legal values for each variable  • When you assign a value to one variable, update legal values  of other variables acardingly  • declare "bad state" when some variable has no legal values left  (Acconsistency  • Create state diagram for all variables -> All arcs chould be consistent  {R, G, B3} {R, B3} {B}		Improvements of DFS for CSP	
Ex: N- Queens Problem - Place I Queen in a spot, then place 2nd Queen.  If 2nd Queen or invalid, "backtrack" and place it again in different spot  ② Value Ordering  · Change order of values [i.e x=1 on left branch, x=0 on right branch]  · this permites the leaf modes, can help find solution quicker  · value ordering heuristic - assign value for the variable with the most restraints (variable with forcest logal values) first  ③ Vanable Ordering  · change order of variables [i.e instead of x ≥ y > z, do x > z > y]  · this primes the tree so there's less to search through  · variable ordering heuristic - assign value for the variable with the least restraints (variable with most legal values) first  ④ Forward Checking  · Maintain a set of legal values for each variable  · When you assym a value to one variable, update legal values of other variables accordingly  · declare "bad state" when some variable has no legal values left  ⑤ Arc Consistency  · Create state diogram for all variables ⇒ fill arcs chould be consistent  {R,G,B} {R,B} {B}		1 Backtrack Search - DFS, but only generate feasible children	
Ex: N. Queens Problem. Place I Queen in a spot, topen place 2 and Queen!  If 2nd Queen is invalid, "backtiack" and place it again in different spot  (Nature Ordering)  · Change order of values [s.e. x+1 on left branch, x=0 on right branch]  · this permites the leaf nodes, can help find solution quecker  · value ordering heuristic - assign value for the variable with A  the most restraints (variable with forcest logal values) first  (Nanable Ordering)  · change order of variables [s.e. Instead of x+y+z, do x+z+y+]  · this prohes the tree so there's less to search through  · variable ordering heuristic - assign value for the variable with  the least restraints (variable with most legal values) first  (Promord Checking)  · Mannain a set of legal values for each variable  · When you assign a value to one variable, update legal values  of of ther variables accordingly  · declare "bad state" when some variable has no legal values left  (R,G,B3) {R,B3} {B}		13 "feasible" - No single constraint is violated	
If 2th Queen is invalid, "backtrack" and place it again in different spot  (Nature Ordering)  (Change order of values [s.e. x=1 on left branch, x=0 on right branch]  (this permutes the leaf nodes, can help find solution quicker)  (value ordering heuristic - assign value for the variable with the most restraints (variable with forcest logal values) first  (Vanable Ordering)  (change order of variables [s.e. instead of x > y > 2, do x > z > y]  (this primes the tree so there's less to search through  variable ordering heuristic - assign value for the variable with the least restraints (variable with most legal values) first  (Forward Checking)  (Maintain a set of legal values for each variable)  (Men you assign a value to one variable, update legal values of other variables accordingly)  (declare "bad state" when some variable has no legal values left  (Arc Consistency)  (Create state diagram for all variables > All arcs should be consistent  {R, G, B} {R, B} {B}		· Ex: N-Queens Problem - Place I Queen in a spot, then place 2nd Queen.	
**Change order of values [s.e x=1 on left branch, x=0 on right branch]  * this permites the leaf modes, can help find solution quecker  * value ordering heuristic - assign value for the variable with the most restraints (variable with fowest legal values) first  **Namable Ordering*  * change order of variables [s.e instead of x+y>z, do x+z>y]  * this prunes the tree so there's less to search through  * variable ordering heuristic - assign value for the variable with the least restraints (variable with most legal values) first  **Promore Checking*  * Maintain a set of legal values for each variable.  * When you assign a value to one variable, update legal values of other variables accordingly  * declare "bad state" when some variable has no legal values left  **Arc Consistency*  **Create state diagram for all variables > All arcs chould be consistent  **\{R,G,B\} \{R,B\} \{B\}\$		If 2nd Queen o invalid, "backtrack" and place it again in different	pat
· this permites the leaf modes, can help find solution quicker  · this permites the leaf modes, can help find solution quicker  · value ordering heuristic - assign value for the variable with the house restraints (variable with forest legal values) first  ③ Variable Ordering  · change order of variables [i.e. instead of x > y > z, do x > z > y]  · this primes the time so there's less to search through  · variable ordering heuristic - assign value for the variable with the least restraints (variable with most legal values) first  ④ Forward Checking  · Maintain a set of legal values for each variable  · When you assign a value to one variable, update legal values of other variables accordingly  · declare "bad state" when some variable has no legal values left  ⑤ Arc (ansistency  · Create state diagram for all variables > All arcs should be consistent  {R,G,B} {R,B} {B}	[ 3 gre	@ Value Ordering . I specified an entire of the entires of policy or	
this permites the leaf nodes, can help find solution quecker  value ordering heuristic - assign value for the variable with A  the most restraints (variable with fowest legal values) first  (available Ordering)  change order of variables [1.e instead of x > y > z, do x > z > y]  this primes the tree so there's less to search through  variable ordering heuristic assign value for the variable with  the least restraints (variable with most legal values) first  (a)  Forward Checking  Maintain a set of legal values for each variable  when you assign a value to one variable, update legal values  of other variables accordingly  declare "bad state" when some variable has no legal values left  (a)  Arc Consistency  (create state diagram for all variables > fill arcs chould be consistent  {R,G,B} {R,B} {B}		· Change order of values [i.e x=1 on left branch, x=0 on right branch]	
* Value ordering heuristic - assign value for the variable with the most restraints (variable with forcest legal values) first  3 Variable Ordering  * Change order of variables [1.e Instead of x > y > z, do x > z > y]  * this prohes the time so there's less to search through  * variable ordering heuristic - assign value for the variable with  the least restraints (variable with most legal values) first  4 Forward Checking  * Maintain a set of legal values for each variable  * When you assign a value to one variable, update legal values  of of her variables accordingly  * declare "bad state" when some variable has no legal values left  5 Arc Consistency  * Create state diagram for all variables > fill arcs chould be consistent  {R,G,B} {R,B} {B}		this permutes the leaf nodes, can help find solution quicker	
The most restraints (variable with fowest logal valves) first  (3) Variable Ordering  (change order of variables [1.e instead of x + y > z, do x + z > y)  (this primes the tree so there's less to search through  variable ordering heuristic assign valve for the variable with  the least restraints (variable with most legal valves) first  (3) Forward Checking  (maintain a set of legal valves for each variable  (when you assign a valve to one variable, update legal valves)  of other variables accordingly  (declare "bad state" when some variable has no legal valves left  (3) Arc Consistency  (create state diagram for all variables > All arcs should be consistent  {R,G,B} {R,B} {B}		· value ordering heuristic - assign value for the variable with	
• this primes the tree so there's less to search through  • variable ordering heuristic - assign value for the variable with  the least restraints (variable with must legal values) first  (Promord Checking  • Maintain a set of legal values for each variable  • When you assign a value to one variable, update legal values  of other variables accordingly  • declare "bad state" when some variable has no legal values left  (Shic Consistency  • Create state diagram for all variables > All arcs should be consistent  {R,G,B} {R,B} {B}		the most restraints (variable with fowest logal valves) first	
• this primes the tree so there's less to search through  • variable ordering heuristic - assign value for the variable with  the least restraints (variable with must legal values) first  (Promord Checking  • Maintain a set of legal values for each variable  • When you assign a value to one variable, update legal values  of other variables accordingly  • declare "bad state" when some variable has no legal values left  (Shic Consistency  • Create state diagram for all variables > All arcs should be consistent  {R,G,B} {R,B} {B}		3 Variable Ordering	
• this primes the tree so there's less to search through  • variable ordering heuristic - assign value for the variable with  the least restraints (variable with must legal values) first  (Promord Checking  • Maintain a set of legal values for each variable  • When you assign a value to one variable, update legal values  of other variables accordingly  • declare "bad state" when some variable has no legal values left  (Shic Consistency  • Create state diagram for all variables > All arcs should be consistent  {R,G,B} {R,B} {B}		· change order of variables [1.e instead of x=y=z, dex=z=y]	
tix least restraints (variable with most legal values) first  (B) Forward Checking  Maintain a set of legal values for each variable  When you assyn a value to one variable, update legal values  of other variables accordingly  declare "bad state" when some variable has no legal values left  (B) Aic Consistency  Create state diagram for all variables > All aics should be consistent  {R,G,B} {R,B} {B}		others prunes the tree so there's less to search through	10000000000000000000000000000000000000
BForward Checking  Maintain a set of legal values for each variable  When you assyn a value to one variable, update legal values  of other variables accordingly  declare "bad state" when some variable has no legal values left  Arc Consistency  Create state diagram for all variables > All arcs should be consistent  {R,G,B} {R,B} {B}		· variable ordering heuristic - assign valve for the variable with	
· Maintain a set of legal values for each variable  · When you assyn a value to one variable, update legal values  of other variables accordingly  · declare "bad state" when some variable has no legal values left  (a) Aic Consistency  · Create state diagram for all variables > All aics should be consistent  {R,G,B} {R,B} {B}			
· When you assyn a value to one variable, update legal values  of other variables accordingly  · declare "bad state" when some variable has no legal values left  (a) Arc Consistency  · Create state diagram for all variables -> All arcs should be consistent  {R,G,B} {R,B} {B}			
of other variables accordingly  * declare "bad state" when some variable has no legal values left  (3) Arc Consistency  * Create state diagram for all variables > All arcs should be consistent  {R,G,B} {R,B} {B}			
· declare "bad state" when some variable has no legal valves left  (3) Arc Consistency · Create state diagram for all variables > All arcs should be consistent  (R,G,B) {R,B} {B}			
3 Arc Consistency  · Create state diagram for all variables → All arcs should be consistent  {R,G,B} {R,B} {B}   △ each value has an			U
· Create state diagram for all variables -> All arcs should be consistent  {R,G,B} {R,B} {B} \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \	(0° v		
{R,G,B} {R,B} {B}			
	(GTAA)		
NNW O VCAI			
ARC 2: {R > B} {G > R, G > B} {B > R} [R, G, B have options > consistent!]			
ARC1: {R>83 {B>??} [Bhas no option, ARE 1 is inconsistent]		ARC1: {R > 83 { B > ?? } LB has no option, ARE 1 is inconsisten	+ [ ]
· make ARCI consistent by removing B from NSW Ethis makes ARCZ inconsistent		· make ARC I consistent by removing B from NSW Lthis makes ARC 2	inconsistent
· make ARC 2 consistent again by removing R from V * of orce on the re-check as an		· make ARC 2 consistent again by removing R from V 12121	16-6.60- 14- 14-14-0
· Complexity: variable has at most "d" values [cardinality], in variables => O(n2 d2 d)		· Complexity: variable has at most "d values Leardinglity), n variables => Ola-d-d	,

2 P Games	Deterministic - Player's possible moves restricted by current state of the game
	Chance - Player's possible moves are determined by chance (ie dice roll)
	Perfect Information - Both players can see full state of thegame
	Imperfect Information - Players can't see full state of the game
	Game Tree - Maps out the possibilities for the game state based and
	Each player's tern - used to try and find the optimal more
	that gives you the best chance of vulnning
	- ply - one person's form more - both persons' turns
And the second second second	Minimax Algorithm - used to find the optimal move for a player,
	assuming the opponent is playing perfectly [uses backtracking DFS logic]
1 %	· A takes max value among child modes, V takes uni value among child modes
	L> O(b") time, O(bm) space, complete, optimal
	Alpha-Beta Pruning - prune off branches that don't need to be explored
	because we already have a better alternative
	40(bd2) time - better than BFS!
kas	proof the plan the paint of the health of the residue was to each as provided and
KRR	Logical Connectives 1 V => (=> )
	and of not rough equivalent
	Conjunction (AND): S, ASZ [conjuncts]
CSP	Disjunction (OR): S. VSz [disjuncts]
	Implication: Si = Sz [antecedent => consequent]
	Contrapositive: [P=>Q] = [Q=>TP]
U	CNF: Clause - disjunction of literals
( ) T	CNF is conjunction of clauses EX: (AVB) 1 (BVCVO)
UT	DNF: term - conjunction of literals
T .	DNF is disjunction of terms Ex: (AAB) V (AATC) V (AATD)
T	Horn: clause with at most one positive literal
44	Note: this is a subset of CNF EX: (AUBV7C)
U	NNF: connectives: 1, v, 7 7 only next to variables
7.00	EX: (((7x v7y) ^Z) vw) v7x
Street of the st	Universal (U) - Normal form can express any sentence
11 T	Tractable (T) - can solve SAT in linear time
UT	ONNF: decomposable NNF

5	Implies X Y >> X Y  >> Y  >> Y  >> Y  >> Y  >> Y  > Y
	w Fa -> Esentence a holds in world w]
	La the variable assignments in w Satisfy &
200	M(x)= {w, wz} > [The models of a are w, wz]
the second	Colorano 3 Lordinal 190 to Wi, We satisfy and Lorandized Hall
	$A \Leftrightarrow B [M(A) = M(B)]$
Complete Antonio	· X is inconsistent junsatisfinale/sontradictory [M(x) = {3]
	· K, B are mutually exclosive [M(A) A M(B) = D] has Area A = intersection
	· A => B [M(x) = M(B)] S = "is a subset of" to 19 months
A Marie No.	Complete term - term containing all of the variables
	valid - a sentence is valid if M(x) = W
N 2 2 2	is the man had a secretary to the secretary to a secretary to the secretar
Lagical Inference	knowledge Base (KB): A={A, Az, An}
	Query (a) - a sentence we're investigating
	Check whether AFX [Does A Imply & ?] 0
	1) Inference Method 1 - Enemerating Models
	· create Truth Table (T/F) for all of the variables
425.00	· create columns for the sentences /knowledge base, these are the outputs
	· fill out contents of truth table.
	· M(A) represents worlds final satisfy statements in KB
	· M(a) represents worlds that satisfy the query &
	· If M(a) & M(a), then a Fa.
	Refutation Theorem: (a holds in D) iff (M(D) = {})
	3 Inference Method 2 - Inference Rules
	Modus Poneas: $\frac{\alpha, \alpha \Rightarrow \beta}{\beta}$ OR-Introduction: $\frac{\alpha, \beta}{\alpha \vee \beta}$ AND-Introduction: $\frac{\alpha, \beta}{\alpha \vee \beta}$
	OR-Introduction: ave AND-Introduction: NAB
	Δ to α → [ α can be inferred from Δ using inference roles R]
(12)	· Soundness: If Atax, Then Atax I have about all stageness.
	1) If we can derive a from A very inference rules R, then a logically follows from A.
(21)	· Completeness: If AFX, Then Atax
	4) If X log roully follows from A, first me can derive & from A using inference rules R.
	EX: D= {AAB, AAB => D3 delle plate & returned?
sall to I ha	$R = \frac{R \cdot R}{\beta}$
	$\Delta' = \{A \land B, A \land B \Rightarrow D, D\}$

Z.E. Europea	Resolution - convert a KB into CNF, cancel out variables between disjunctions to
	there are create new disjunctions and assessed as
	· Refutation complete - Resolution leads to contradiction of CNF is unsatisfiable
	· Unit Resolutions (Linear Time) faster than normal Resolution (Exponential Time)
	La however, unit resolutions aren't refutation complete Ideficing rules with one remarks
	1 List Given Statements: Convert KB and claim of into CNF, list
Peth street.	each individual disjunction as a rule
	(2) Cancel out yariables from given rules to create new rules
	until no more rules can be applied (each variable is by itself in a rule)
	<ul> <li>If there was a contradiction - Δ ^ α is unsat, but we prove α</li> </ul>
	· If there was no contradiction - Δ^ α is sat, but we can't prove a
	Conversion to CNF
	1) Gret rid of all connectives except A, V, 7
	$\alpha \propto \beta \rightarrow \gamma \propto V \beta$
	· a cos o (a cos) A (B cox) -> (Tave) A (Teva)
	2 Push negations inwards in a decided and the second and the secon
All da	· T(a AB) -> (Ta VTB) · T(aVB) -> (Ta ATB)
	3 Distribute V over A of Adjusted and Adjust
	$\circ (\alpha \wedge \beta) \vee \mathcal{F} \longrightarrow (\alpha \vee \mathcal{F}) \wedge (\beta \vee \mathcal{F})$
	3) Inference Method 3: Reduction to SAT
	· A Fa iff Anix is unsaf
	· A -> & [B 17x is unsat]
U	$M \cdot \Delta \Leftrightarrow \alpha [\Delta \Rightarrow \alpha; \alpha \Rightarrow \Delta]$
	· D 17x is unsat; 7 D 1 x ts unsat
UT	· Ars valid [ Ars unsat]
	· D and a partually exclusive [ D and A unsut]
The state of	· Complete Methods - will return if something is SAT or not
A male	· DPLL - specialized backtracking algorithm, basically uses
	Resolution to determine if we can prime the tree
A con bods	· Incomplete Methods - Local Search
	· Start with a state. if it's a solution, stop there.
	· If it's not a solution, change value of one variable (neighbor) Repeat until solution.
	THAT ALL THE SEE TO GO SA A BAAS A

5	
(min-conflicts)	· idea is to go to a meighbor that is closer to a solution, keep repealing
(hill-climbing)	· local minimum - all neighbors are worse, restart at different point
	· all neighbors are basically same as current state - go to a random one
	· Local Search has no memory, possible to repeat states
1	· Simulated Annewling - traverse to neighbor based on prabability
	S: current state N: neighbor DE = violations(n) - violations(s)
	If DE <0: Traverse to that no plantill at month of
	Else: traverse to n based on probability [ederr]
	(the more violations a neighbor has, the less likely I go to it)
	T: as I get deeper into the glaph, Tincreases
	La micre cautious about choosing had norghburs the firsther wego
	9 Inference Method 4: Tractable Circuits
	. # SAT - doesn't solve SAT, counts the number of satisfiable assignments
	· NNF Creatt
	· Decomposability - direct children of an AND gate can't use the same variables (negations ok)
Solve #SAT	· Determinism - all combinations of pairs of children for an OR gate
you linear time	must cancel each other out
	· ORgates must have at most i true input under any circuit input
-	Smoothness-direct children of an ORgate must use the same Variables (negations ok)
	How many assignments satisfy the problem (#SAT)?
	() Assign all leaves to be   [literal T = 1, literal F=0]
	3 Traverse upwards, multiplying at AND, adding at OR gates
	3 Result at the root is the result of #SAT
	Carry and the second se
	(A top second se
	Consiste Consistence of the Constant of the Co
	The same state of the same sta
	. Mar Abalta da sur su torre sur la la sur sur la s
	a country to the second
	The Commercial Control of the Contro

.000	
M	
4	O Do the first triangle normally, no proving involved, Save that value as X.
940	2) If triangle to the right is 1:
	look through its children, if there's a child = x then change
	that one and prine all the ones after it (brigger than smallestone)
	If triangle to the right is V:
	look through its dilbren. If there's a child & X then choose
	that one and prime all the ones after it (smaller than biggestone),
	3 Continue on the row or bove until finished.
	The state of the s
6	and wealth of the comment of the delication of the contract of
	O Interest of Mathe of A Presidents streams and an amount
e) essex	AND AND AND AND ADMINISTRATION OF AND ASSESSED ASSESSED ASSESSED ASSESSED ASSESSED.
	A MANY CHOOSE A SECOND AND A SECOND ASSECTION AS SECOND ASSECTION ASS
The market March	Commenced the property of the state of the s
	Sinher & many all the ship to the same of a contraction in the recommended. I make subse
	A TORREST OF THE STATE OF THE S
tagan	Charles from the house of the first that the second through the second
(As continued of	donal was a ser but the stand of any the west to sent the Andrech recording .
	The course of the first was the constitution with the for it many more to grant with
	O Frank all leaves to be I federal of the author to
	Traverse against the following and the same of the sam
	Charles to the seal of the sea
	- B Cla D cese C C B Ch de Server
	The Armount Car would be a second to the sec
	The Broad a protection call when I have not allowed
	The plant plant the trade of the second of t
	a state to a great which has been been a supported that the same of the same of
	Regulation to distance of the second are not the fines.
	2 La managela Mr. M. a fillulas - Consort Comme de Conso
	the state of the secretaries of entities tolleged the state of the second
	- If it is not a souther change in addition or make ingular to great the control of the control