Planning as Satisfiability using Plan Graphs

Written by: Henry Kautz, David McAllester, and Bart Selman
Presented by: Wyatt Scott

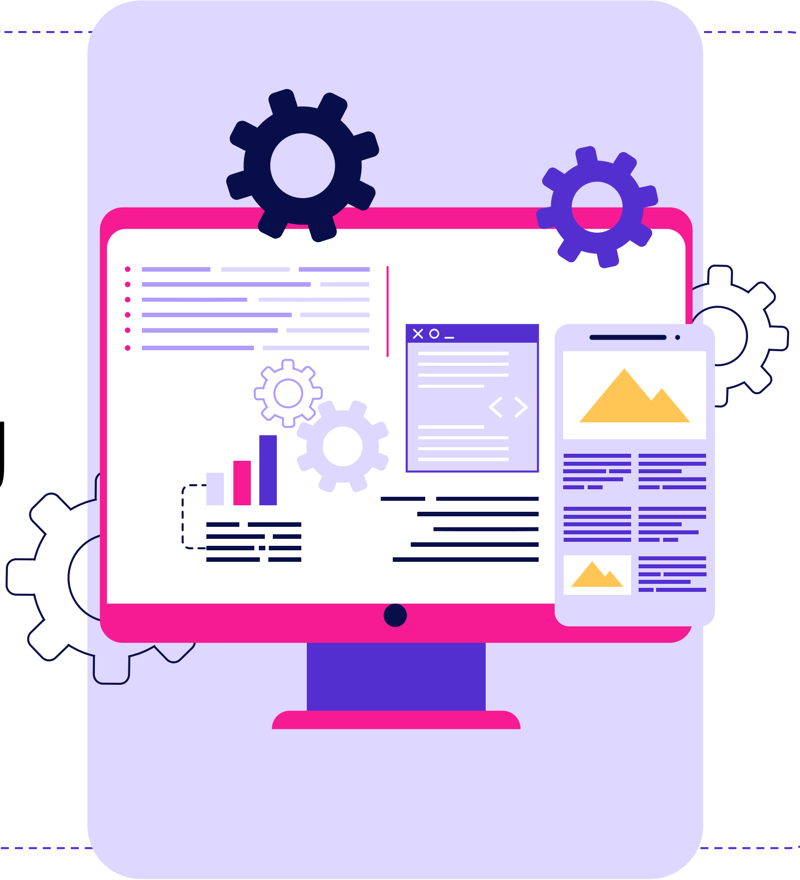


TABLE OF CONTENTS

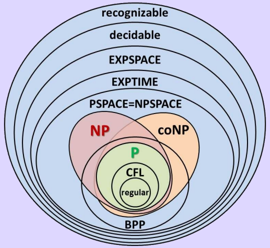
01 Introduction

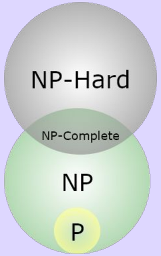
02 Review of Graphplan

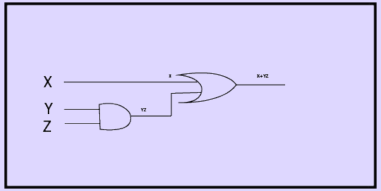
03 Review of SAT

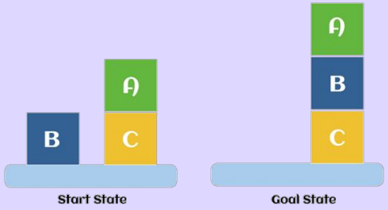
04 General Framework

05 Encodings









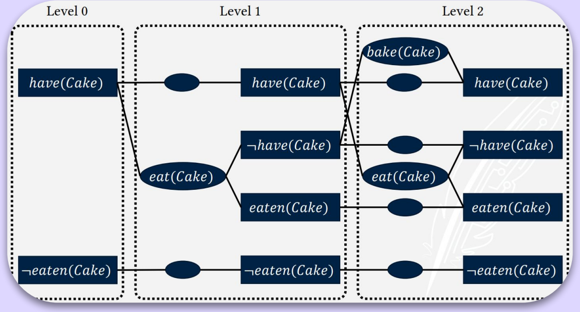
Word planning problem

01

Introduction

Introduction

- Planning problems can be efficiently solved by SAT algorithms
- A polynomial reduction from planning to SAT is always possible
  - However, it is not always practical
  - $O(n^3)$ size increase
  - $O(n^4)$ size increase
- Planning graphs could be interpreted as propositional CNF formulas
  - Different routes of reducing are available



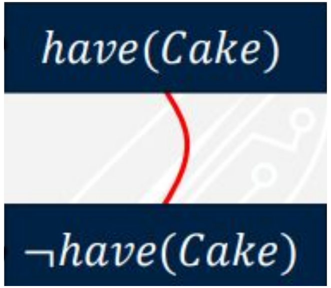
02

Review of Graphplan

GraphPlan Structure

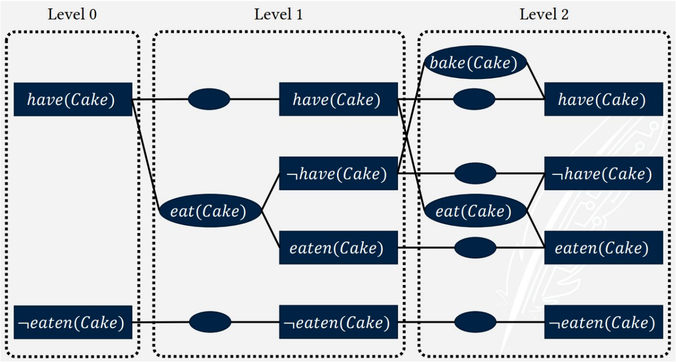
- Nodes
  - Literal node
    - A single ground predicate literal
  - Step node
    - A ground action
- Mutexes
  - Define which steps cannot be taken at the same level

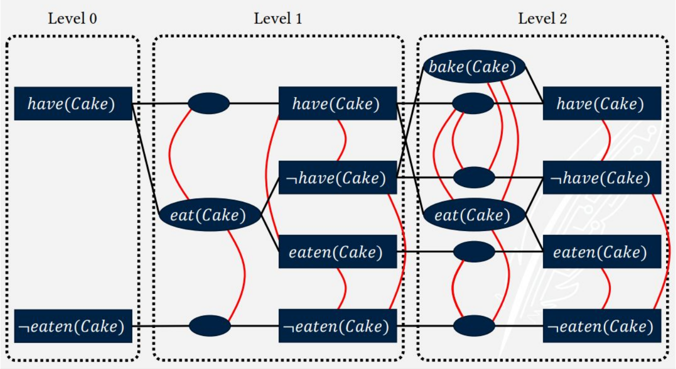
$have(Cake)$

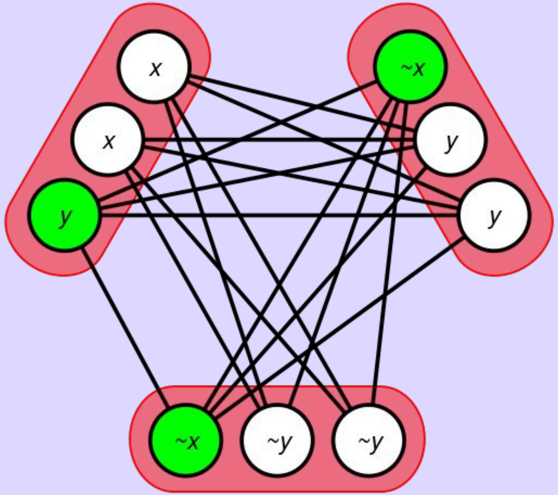


Solving a Graphplan

1. Set G to be the list of goals, and n to be the last level in the graph
2. If n = 0, return the plan as a solution
3. Choose a set of steps S which achieve all goals in G
   a. Every set of steps must not be mutexed
4. Add all the steps in S to the plan
5. Let G be the set of all preconditions of the steps in S
6. Return to step 2 with n = n – 1
7. If a plan cannot be found, add a level to the graph and start back at step 1





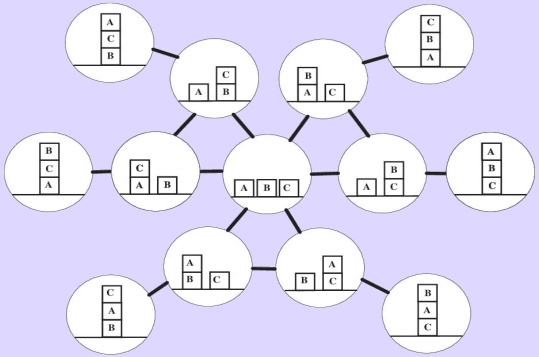


03

Review of SAT

SAT

- Conjunctive Normal Form (CNF)
  - Ex: $(x \lor \neg y) \land (\neg y \lor \neg z) \land (\neg x \lor z)$
- Every variable must be assigned a truth value
  - In the above example, x = true, y = false, and z = true, satisfies the expression
- SAT is NP-complete



04

General Framework

Definitions

- Defined using standard STRIPS notation
  - Ops – A set of operator definitions
    - Defined by Preconditions, an Add List, and Delete List
    - (A Delete List is not needed if negations are used)
  - Dom – A domain of individuals
  - S_0 – Initial State
  - S_1 – Final State

Move(x, y, z)
PRE: CLEAR(x), ON(x, y), CLEAR(z)
ADD: CLEAR(y), ON(x, z)
DEL: CLEAR(z), ON(x, y)

Definitions

```
Move(x, y, z)
        PRE:  CLEAR(x), ON(x, y), CLEAR(z)
        ADD: CLEAR(y), ON(x, z)
        DEL: CLEAR(z), ON(x, y)
```

Action – The instantiation of an operator over a domain.

Ex: Move(x, y, z)

Fluent – The instantiation of a predicate over a domain.

Ex: On(x, y)

Finding a Solution

- **A sequence of actions defines a solution to a problem if:**
  - It transforms the initial state into a superset of the goal state
  - The preconditions of each action appear in the current state
  - No action both adds and deletes the same fluent
- **The problem can also be bounded by a maximum number of levels so it will never enter an infinite loop when there is no solution**
  - Number of Actions = $(Ops)*(Dom)^{AOps}$
  - Number of Fluents = $(Pred)*(Dom)^{APred}$

05

Encodings

Linear Encodings

- Adds a time parameter
  - Excludes unintended models that are problematic for SAT
- The reduction yields the following clauses
  - The Closed World Assumption holds (what is not true is false)
  - If an action holds at time i:
    - Its preconditions hold at time i
    - Its added fluents hold at time i + 1
    - The negation of each of its deleted fluents hold at time i + 1
  - The Classical Frame condition holds for all actions
    - Fluents which aren't affected by an action remain constant
  - Exactly one action occurs at each time instant

Linear Encodings - Operator Splitting

- **A reduction to SAT is not always feasible**
  - The difficulty is dominated by the number of arguments of an operator
- **One resolution is to split up actions**

Move(A, B, C)
PRE: CLEAR(A), ON(A, B), CLEAR(C)
ADD: CLEAR(B), ON(A, C)
DEL: CLEAR(C), ON(A, B)

Source(A, 3) $\wedge$ Object(B, 3) $\wedge$ Destination(C, 3)

Linear Encodings - Explanatory Frame Axioms

- **If the truth value of a fluent changes, then an add or delete occurred**
- **If none of those actions occurred, then the fluent stays the same**
  - Results in null actions not being needed
- **This creates a frame axiom which has less clauses, but each clause is longer**
  - In the worst case, the resulting formula is the same size as with operator splitting

Parallelized Encodings

- Multiple actions can occur at the same time step
    - Reduces the size of the encoding
- First creates a partial order
- Creates a total order at the end

Parallelized Encodings - Graphplan Based

- **Operators imply their preconditions**

$\boxed{Load(A, R, L, 2) \rightarrow (At(A, L, 1) \wedge At(R, L, 1))}$

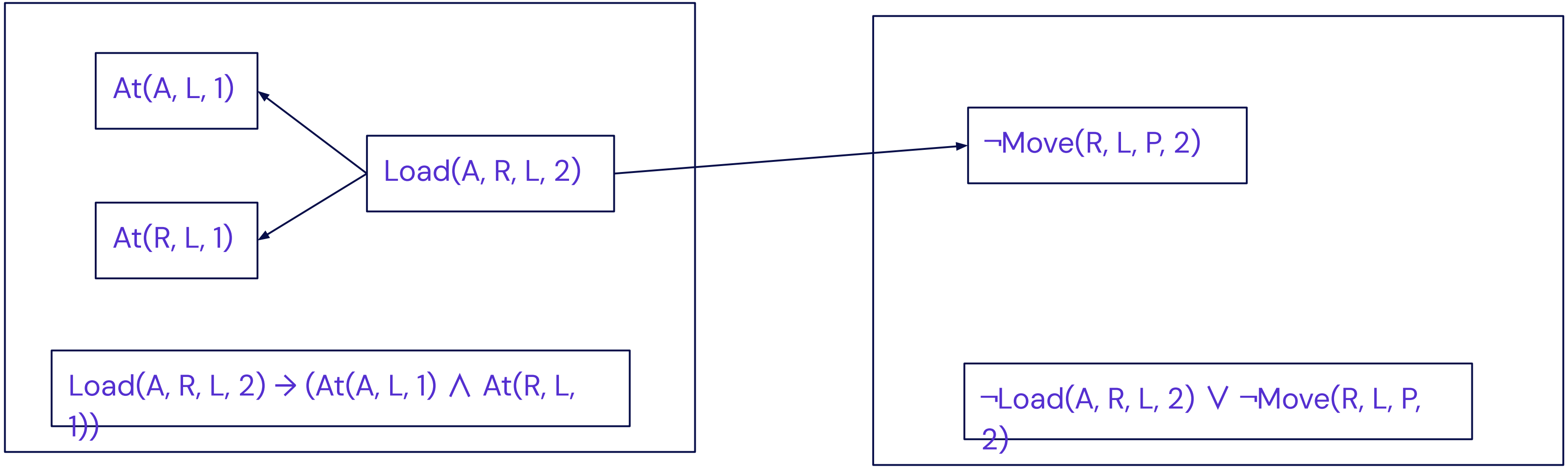
- **Each fact implies it's preconditions**

$\boxed{In(A, R, 3) \rightarrow (Load(A, R, L, 2) \vee Load(A, R, P, 2) \vee Maintain(In(A, R), 2))}$

- **Conflicting actions are mutually exclusive**

$\boxed{\neg Load(A, R, L, 2) \vee \neg Move(R, L, P, 2)}$

Parallelized Encodings - Graphplan Based



Lifted Encodings

- **Provides the best result asymptotically**
- **$|S_0| = |S_G| = |Dom| = n$**
  - $O(n^2)$ Boolean Variables
  - $O(n^3)$ Literal Occurrences
- **The Lifted SAT problem**
  - An NP-Complete problem more general than SAT

Example - Sussman Anomaly

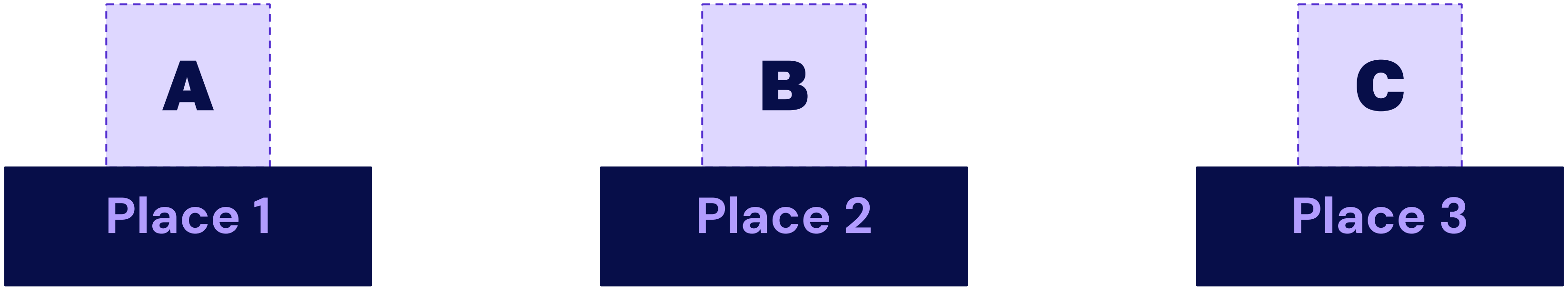
Move(A, B, C)
PRE: CLEAR(A), ON(A, B), CLEAR(C)
ADD: CLEAR(B), ON(A, C)
DEL: CLEAR(C), ON(A, B)

Goal: A on B ∧ B on C

Example - Sussman Anomaly

Move(A, B, C)
PRE: CLEAR(A), ON(A, B), CLEAR(C)
ADD: CLEAR(B), ON(A, C)
DEL: CLEAR(C), ON(A, B)

Goal: A on B ∧ B on C



Example - Sussman Anomaly

Move(A, B, C)
PRE: CLEAR(A), ON(A, B), CLEAR(C)
ADD: CLEAR(B), ON(A, C)
DEL: CLEAR(C), ON(A, B)

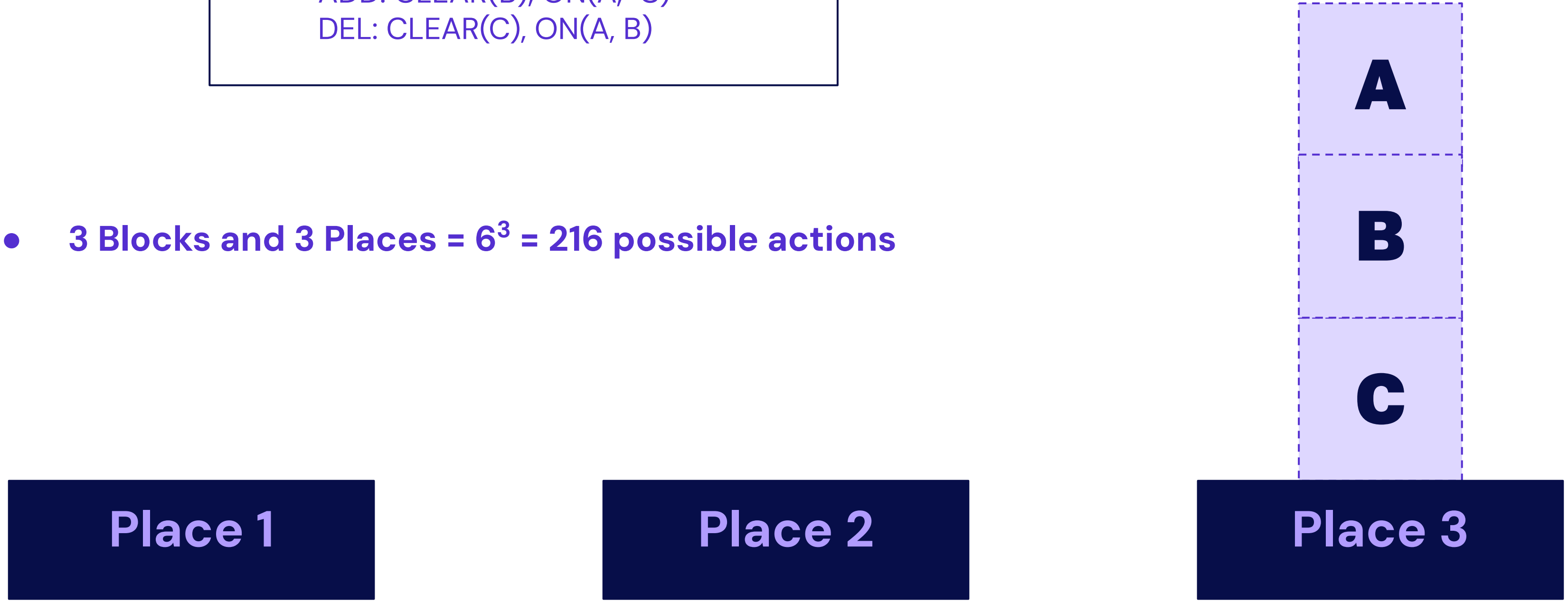
Goal: A on B ∧ B on C

Example - Sussman Anomaly

Move(A, B, C)
PRE: CLEAR(A), ON(A, B), CLEAR(C)
ADD: CLEAR(B), ON(A, C)
DEL: CLEAR(C), ON(A, B)

Goal: A on B $\wedge$ B on C

- **3 Blocks and 3 Places = 6^3 = 216 possible actions**



A Complete Causal Plan

- Ground causal link
  - An assertion of the form $o_i \Phi \rightarrow o_j$
- Complete causal plan
  - An assignment of ground actions to step names
  - A set of causal links
  - A set of step ordering assertions
    - Every prerequisite has a cause
    - Every causal link is true
    - The ordering constraints are consistent (transitive property)
      - If $o_i < o_k$ and $o_k < o_j$, then $o_i < o_j$

Example - Sussman Anomaly

Goal: A on B $\wedge$ B on C

Move(C, A, Place3) $\xrightarrow{\text{Clear(A)}}$ Move(A, Place1, B)

Move(B, Place2, C) $\xrightarrow{\text{On(B, C)}}$ Final

Move(A, Place1, B) $\xrightarrow{\text{On(A, B)}}$ Final

A

B

C

Place 1

Place 2

Place 3

Conclusion

- **Utilized concise SAT encodings**
  - Allowed SAT algorithms to outperform planning systems
- **Polynomial-time reductions from STRIPS planning to CNF formulas**
- **Described lifted causal encodings**

Questions?