Planning and Optimization

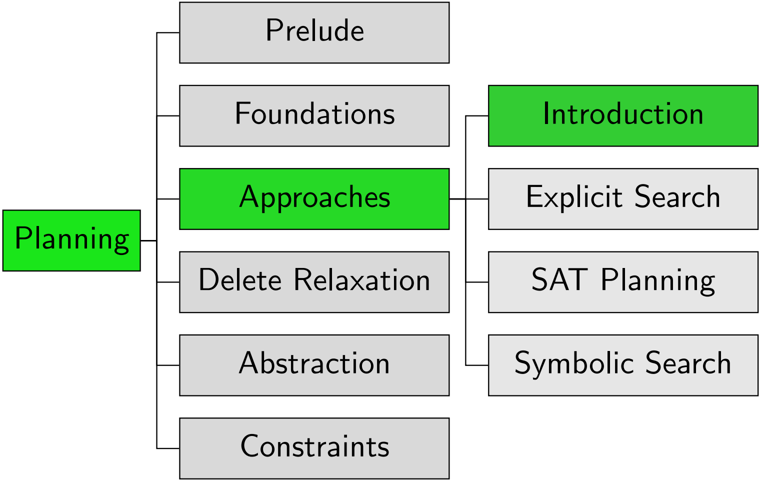
C2. Overview of Classical Planning Algorithms (Part 2)

Malte Helmert and Gabriele Röger

Universität Basel

October 6, 2025

Content of the Course



The Big Three (Repeated from Last Chapter)

Of the many planning approaches, three techniques stand out:

- explicit search $\rightsquigarrow$ Chapters C3–C4, Parts D–F
- SAT planning $\rightsquigarrow$ Chapters C5–C6
- symbolic search $\rightsquigarrow$ Chapters C7–C8

also: many algorithm portfolios

SAT Planning

SAT Planning: Basic Idea

- formalize problem of finding plan with a given horizon (length bound) as a propositional satisfiability problem and feed it to a generic SAT solver
- to obtain a (semi-) complete algorithm, try with increasing horizons until a plan is found (= the formula is satisfiable)
- important optimization: allow applying several non-conflicting operators "at the same time" so that a shorter horizon suffices

SAT Encodings: Variables

- given propositional planning task $\langle V, I, O, \gamma \rangle$
- given horizon $T \in \mathbb{N}_0$

Variables of SAT Encoding

- propositional variables v^i for all $v \in V$, $0 \leq i \leq T$ encode state after i steps of the plan
- propositional variables o^i for all $o \in O$, $1 \leq i \leq T$ encode operator(s) applied in i-th step of the plan

Design Choice: SAT Encoding

Again, there are several important design choices.

Design Choice: SAT Encoding

- sequential or parallel
- many ways of modeling planning semantics in logic

$\rightsquigarrow$ main focus of research on SAT planning

Design Choice: SAT Solver

Again, there are several important design choices.

Design Choice: SAT Solver

- out-of-the-box like Glucose, CaDiCal, MiniSAT
- planning-specific modifications

Design Choice: Evaluation Strategy

Again, there are several important design choices.

Design Choice: Evaluation Strategy

- always advance horizon by $+1$ or more aggressively
- possibly probe multiple horizons concurrently

Symbolic Search

Symbolic Search Planning: Basic Ideas

- search processes sets of states at a time
- operators, goal states, state sets reachable with a given cost etc. represented by binary decision diagrams (BDDs) (or similar data structures)
- hope: exponentially large state sets can be represented as polynomially sized BDDs, which can be efficiently processed
- perform symbolic breadth-first search (or something more sophisticated) on these set representations

Symbolic Breadth-First Progression Search

prototypical algorithm:

Symbolic Breadth-First Progression Search

def bfs-progression(V, I, O, γ):
    $goal\_states := models(\gamma)$
    $reached_0 := \{I\}$
    $i := 0$
    loop:
        if $reached_i \cap goal\_states \neq \emptyset$:
            return solution found
        $reached_{i+1} := reached_i \cup apply(reached_i, O)$
        if $reached_{i+1} = reached_i$:
            return no solution exists
        $i := i + 1$

Symbolic Breadth-First Progression Search

prototypical algorithm:

Symbolic Breadth-First Progression Search

def bfs-progression(V, I, O, γ):
  $goal\_states := models(\gamma)$
  $reached_0 := \{I\}$
  $i := 0$
  loop:
    if $reached_i \cap goal\_states \neq \emptyset$:
      return solution found
    $reached_{i+1} := reached_i \cup apply(reached_i, O)$
    if $reached_{i+1} = reached_i$:
      return no solution exists
    $i := i + 1$

$\rightsquigarrow$ If we can implement operations *models*, $\{I\}$, $\cap$, $\neq \emptyset$, $\cup$, *apply* and $=$ efficiently, this is a reasonable algorithm.

Design Choice: Symbolic Data Structure

Again, there are several important design choices.

Design Choice: Symbolic Data Structure

- BDDs
- ADDs
- EVMDDs
- SDDs

Other Design Choices

- additionally, same design choices as for explicit search:
  - search direction
  - search algorithm
  - search control (incl. heuristics)
- in practice, hard to make heuristics and other advanced search control efficient for symbolic search $\rightsquigarrow$ rarely used

Planning System Examples

Planning Systems: FF

FF (Hoffmann & Nebel, 2001)

- problem class: satisficing
- algorithm class: explicit search
- search direction: forward search
- search algorithm: enforced hill-climbing
- heuristic: FF heuristic (inadmissible)
- other aspects: helpful action pruning; goal agenda manager

$\rightsquigarrow$ breakthrough for heuristic search planning;
winner of IPC 2000

Planning Systems: LAMA

LAMA (Richter & Westphal, 2008)

- problem class: satisficing
- algorithm class: explicit search
- search direction: forward search
- search algorithm: restarting Weighted A* (anytime)
- heuristic: FF heuristic and landmark heuristic (inadmissible)
- other aspects: preferred operators; deferred heuristic evaluation; multi-queue search

⇝ still one of the leading satisficing planners;
winner of IPC 2008 and IPC 2011 (satisficing tracks)

Planning Systems: Madagascar-pC

Madagascar (Rintanen, 2014)

- problem class: satisficing
- algorithm class: SAT planning
- encoding: parallel ∃-step encoding
- SAT solver: using planning-specific action variable selection
- evaluation strategy: exponential horizons, parallelized probing
- other aspects: invariants

$\rightsquigarrow$ second place at IPC 2014 (agile track)

Planning Systems: SymBA*

SymBA* (Torralba, 2015)

- problem class: optimal
- algorithm class: symbolic search
- symbolic data structure: BDDs
- search direction: bidirectional
- search algorithm: mixture of (symbolic) Dijkstra and A*
- heuristic: perimeter abstractions/blind

⇝ winner of IPC 2014 (optimal track)

Planning Systems: Scorpion

Scorpion 2023 (Seipp, 2023)

- problem class: optimal
- algorithm class: explicit search
- search direction: forward search
- search algorithm: A^*
- heuristic: abstraction heuristics and cost partitioning

$\rightsquigarrow$ runner-up of IPC 2023 (optimal track)

Planning Systems: Fast Downward Stone Soup

Fast Downward Stone Soup 2023, optimal version (Büchner et al., 2023)

- problem class: optimal
- algorithm class: (portfolio of) explicit search
- search direction: forward search
- search algorithm: A^*
- heuristic: all admissible heuristics considered in the course

⇝ winner of IPC 2011 (optimal track);
various awards in IPC 2011–2023

Planning Systems: SymK

SymK (Speck et al., 2025)

- problem class: optimal
- algorithm class: symbolic search
- symbolic data structure: BDDs
- search direction: bidirectional
- search algorithm: symbolic Dijkstra algorithm
- heuristic: blind

Summary

Summary

big three classes of algorithms for classical planning:

- explicit search
  - design choices: search direction, search algorithm, search control (incl. heuristics)
- SAT planning
  - design choices: SAT encoding, SAT solver, evaluation strategy
- symbolic search
  - design choices: symbolic data structure + same ones as for explicit search