**Integrating Planning and Scheduling: Status and Prospects**

*Subbarao Kambhampati*

Arizona State University
rakaposhi.eas.asu.edu/yochan.html

---

**Planning vs. Scheduling**

A Continuum

Scheduling
- Set of jobs (may have of tasks in some (partial) order)
- Temporal constraints on jobs
  - EST, LFT, Duration
- Contention constraints
  - Each task can be done on a subset of machines
Find start times for jobs that are optimal (wrt make-spans, resource consumption etc)

Planning
- Initial state & a set of Goals,
  - A library of actions
    - Preconditions/effects
      - Discrete/Continuous
  - Resource requirements
Synthesize a sequence of actions capable of satisfying goals

---

Research into planning and scheduling methods has largely been de-coupled
Need for Integration

- Most existing schedulers concentrate only on resource allocation, ignoring action selection
  - E.g. HSTS operation scheduling

- Most existing planners concentrate on action selection, ignoring resource allocation
  - Plan-based interfaces
  - Interactive decision support

- Many real-world problems require both capabilities
  - Supply Chain Management problems
    » I2, ILOG, Manugistics
  - Planning in domains with durative actions, continuous change
    » NASA RAX experiment

Why now?

- Significant scale-up in plan synthesis in last 4-5 years
  - 5/6 action plans in minutes to 100 action plans in minutes
  - Breakthroughs in search space representation, heuristic and domain-specific

- Significant strides in our understanding of connections between planning and scheduling
  - Rich connections between planning and CSP/SAT/ILP
    » Vanishing separation between planning techniques and scheduling techniques
Approaches for Integration

- Extend schedulers to handle action and resource choices
- Extend planners to deal with resources, durative actions and continuous quantities

Coupled Architectures
- De-coupled
- Loosely Coupled (RealPlan System)

Overview

Why integrate planning and scheduling?
- Planning: The state of the art
- Scheduling: The state of the art
- Integrating Planning and Scheduling
Planning: The State of the Art

Overview
- Why integrate planning and scheduling?
- Planning: The state of the art
- Scheduling: The state of the art
- Integrating Planning and Scheduling

(Deterministic) Planning: The problem

- States are modeled in terms of (binary) state-variables (factored rep.)
  -- Complete initial state, partial goal state
- Actions are modeled as state transformation functions
  -- Syntax: ADL language (Pednault)
- Plans are sequences of actions

**(Deterministic)** Planning: The problem

- States are modeled in terms of (binary) state-variables (factored rep.)
  -- Complete initial state, partial goal state
- Actions are modeled as state transformation functions
  -- Syntax: ADL language (Pednault)
- Plans are sequences of actions
The (too) many brands of classical planners

Planning as Theorem Proving  (Green’s planner)

Planning as Search

Search in the space of States  (progression, regression, MEA)
  (STRIPS, PRODIGY, TOPI, HSP, HSP-R,
   UNPOP, FF)

Search in the space of Plans  (total order, partial order,
  protections, MTC)
  (Interplan, SNLP, TOCL,
   UCPSP, TWEAK)

Planning as CSP/ILP/SAT/BDD
  (Graphplan, IPP, STAN, SATPLAN, BLackBOX, GP-CSP, BDDPlan)

Planning as Model Checking

Search in the space of Plans  (total order, partial order,
  protections, MTC)
  (Interplan, SNLP, TOCL,
   UCPSP, TWEAK)

Planning as Search

Search in the space of States  (progression, regression, MEA)
  (STRIPS, PRODIGY, TOPI, HSP, HSP-R,
   UNPOP, FF)

Planning as Theorem Proving  (Green’s planner)

Refinement Planning: The idea

Narrowing sets of action sequences
  to progress towards sets of solutions

Partial plans

Refinements
Plan Representation

Partial plan = (Steps, Orderings, Aux. Constraints)

P:

0 1: Load(A)

2: Fly()

4: Unload(A) \infty

In(A)@2

At(R,E)

3: Load(B)

Semantics in terms of Candidate sets
--Candidate is an action sequence that satisfies all the plan constraints (but can have additional actions)

Refinements split and prune candidate sets

(The) Three Refinement Strategies

Progression

Regression

Add in the middle

Least Commitment Regression
Tradeoffs among Refinements

FSR and BSR must commit to both position and relevance of actions

+ Give state information (Easier plan validation)
- Leads to premature commitment
- Too many states when actions have durations

Plan-space refinement (PSR) avoids constraining position

+ Reduces commitment (large candidate set/branch)
- Increases plan-validation costs
+ Easily extendible to actions with duration

A flexible Split & Prune search for Refinement Planning

Refineplan( \( \mathcal{P} : \text{Plan} \) )

0*: If \( \langle \mathcal{P} \rangle \) is empty, Fail.
1. If a minimal candidate of \( \mathcal{P} \) is a solution, terminate.
2. Select a refinement strategy \( \mathcal{R} \).
   - Apply \( \mathcal{R} \) to \( \mathcal{P} \) to get a new plan set \( \mathcal{P}' \)
3. Split \( \mathcal{P}' \) into \( \mathcal{P} \) plansets
4. Non-deterministically select one of the plansets \( \mathcal{P}_i \)
   - Call Refine(\( \mathcal{P}_i \))
Broad Themes in the Planning Renaissance

- Disjunctive Representations
- Reachability/Relevance Analysis
- Connections to combinatorial substrates
- Sophisticated domain pre-processing Techniques

---

**Ontable(A)**

**Ontable(B)**

**Clear(A)**

**Clear(B)**

**hand-empty**

**holding(A)**

**holding(B)**

**~Clear(A)**

**~Ontable(A)**

**Ontable(B)**

**~handempty**

**Pickup(A)**

**Pickup(B)**

**Progression Search**

---

**Init:**

Ontable(A), Ontable(B),
Clear(A), Clear(B), hand-empty

**Goal:**

~clear(B), hand-empty

---

**Putdown(A)**

**Stack(A,B)**

**Putdown(B)??**

---

**Tutorial on Recent Advances in AI Planning (UCAI-99, AAAI-00)**

http://naka.post.hkust.edu/planning-tutorial
Optimistic Projection of Reachability

Init:
Ontable(A), Ontable(B),
Clear(A), Clear(B), hand-empty

Goal:
~clear(B), hand-empty
Heuristics based on the Planning Graph

- \( \text{lev}(S) \): index of the first level where all props in \( S \) appear nonmutexed.
  - If there is no such level, then
    - If the graph is grown to level off, then \( \infty \)
    - Else \( k+1 \) (\( k \) is the current length of the graph)

Set-Level heuristic: \( h(S) = \text{lev}(S) \)
Admissible but not very informed

Sum heuristic: \( h(S) = \sum_{p \in S} \text{lev}({p}) \) Inadmissible
Assumes that sub-goals are independent

Adjusted Sum heuristic: [Sanchez et al., 2000]
\[
H_{\text{AdjSum2M}}(S) = \text{length}(\text{RelaxedPlan}(S)) + \max_{p,q \in S} \delta(p,q)
\]
where \( \delta(p,q) = \text{lev}({p,q}) - \max\{\text{lev}(p), \text{lev}(q)\} \)

Planning as Plangraph Solution Extraction

If there exists a \( k \)-length plan, it will be a subgraph of the \( k \)-length planning graph.
Planning is thus searching for a “valid” subgraph of the planning graph.

Combinatorial search.
Can be cast into any combinatorial substrate (e.g. CSP, SAT, ILP...)
(very) Quick overview of CSP/SAT concepts

- Constraint Satisfaction Problem (CSP)
  - Given
    - A set of discrete variables
    - Legal domains for each of the variables
    - A set of constraints on values groups of variables can take
  - Find an assignment of values to all the variables so that none of the constraints are violated
- SAT Problem = CSP with boolean variables
- TCSP = CSP where variables are time points and constraints describe allowed distances

A solution: $x=B, y=C, u=D, v=E, w=D, l=B$

Important ideas in solving CSPs

Variable order heuristics:
Pick the most constrained variable
--Smallest domain, connected to most other variables,
causes most unit propagation,
causes most resource contention,
has the most distance etc...

Value ordering heuristics
Pick the least constraining value

Consistency enforcement
- k-consistency; adaptive consistency. (pre-processing)
- Forward Checking, unit propagation during search (dynamic)

Search/Backtracking
- DDB/EBL: Remember and use interior node failure explanations
- Randomized search
Posing Plangraph Solution Extraction as a CSP/SAT

Variables: literals in proposition lists
Values: actions supporting them
Constraints: Mutex and Activation constraints

Compilation to Integer Linear Programming

ILP: Given a set of real valued variables, a linear objective function on the variables, a set of linear inequalities on the variables, and a set of integrality restrictions on the variables, find the values of the feasible variables for which the objective function attains the maximum value.

- c/z integer programming corresponds closely to SAT problem

Motivations
- Ability to handle numeric quantities, and do optimization
- Heuristic value of the LP relaxation of ILP problems

Conversion
- Convert a SAT/CSP encoding to ILP inequalities
  - E.g. \( X \lor \neg Y \lor Z \Rightarrow x + (1 - y) + z \geq 1 \)
  - Explicitly set up tighter ILP inequalities
    - If \( X, Y, Z \) are pairwise mutex, we can write \( x + y + z \leq 1 \)
      (instead of \( x + y \leq 1 ; y + z \leq 1 ; z + x \leq 1 \))
Relative Tradeoffs Offered by the various compilation substrates

- CSP encodings support implicit representations
  - More compact encodings [Do & Kambhampati, 2000]
  - Easier integration with Scheduling techniques
- ILP encodings support numeric quantities
  - Seamless integration of numeric resource constraints [Walser & Kautz, 1999]
  - Not competitive with CSP/SAT for problems without numeric constraints
- SAT encodings support axioms in propositional logic form
  - May be more natural to add (for whom ;-)?
- BDDs are very popular in CAD community
  - Commercial interest may spur effective algorithms (which we can use)

Disjunctive Planning

- Idea: Consider Partial plans with disjunctive step, ordering, and auxiliary constraints
- Motivation: Provides a lifted search space, avoids re-generating the same failures multiple times (also, rich connections to combinatorial problems)
- Issues:
  - Refining disjunctive plans
    » Graphplan (Blum & Furst, 95)
  - Solution extraction in disjunctive plans
    » Direct combinatorial search
    » Compilation to CSP/SAT/ILP
Planning: Current Status

- Disjunctive planners as well as heuristic state-search planners scale well
  - Plans with up to 100 actions synthesized in minutes
  - Impressive performances in AIPS-1998 and AIPS-2000 Competitions
- Plan-space planners with comparable performance are still to be developed [Long & Kambhampati, IJCAI-2001 (may be)]

Scheduling: The State of the Art

Overview

- Why integrate planning and scheduling?
- Planning: The state of the art
- Scheduling: The state of the art
- Integrating Planning and Scheduling
**Scheduling: Brief Overview**

Jobshop scheduling
- Set of jobs
  - Each job consists of tasks in some (partial) order
- Temporal constraints on jobs
  - EST, LFT, Duration
- Contention constraints
  - Each task can be done on a subset of machines

**CSP Models**
- Time point model
  - Tasks as variables, Time points as values
  - EST, LFT, Machine contention as constraints
- Inter-task precedences as variables (PCP model)

**CSP Techniques**
- Customized consistency enforcement techniques
  - ARC-B consistency
  - Edge-finding
- Customized variable/value ordering heuristics
  - Contention-based
  - Slack-based
- MaxCSP; B&B searches

---

**Job Shop Scheduling as a CSP**

**Jobs**

<table>
<thead>
<tr>
<th>J_1</th>
<th>O_i/R_i</th>
<th>O_i/R_j</th>
<th>O_j/R_k</th>
</tr>
</thead>
<tbody>
<tr>
<td>J_2</td>
<td>O_i/R_i</td>
<td>O_j/R_k</td>
<td></td>
</tr>
<tr>
<td>J_3</td>
<td>O_i/R_i</td>
<td>O_j/R_k</td>
<td></td>
</tr>
</tbody>
</table>

**Start Point Representation**

Variables: Start time st_i
Domain: [est_i, lst_i]
Precedence constraints:
\[ st_i + du_i \leq st_j \]
Capacity Constraints:
\[ st_i + du_i \leq st_i \lor st_i + du_i \leq st_j \]

**PCP Representation**

Variables: Ordering(i,j,R) for task i and j contending for resource R.
Domain: \{i-before-j, j-before-i\}
Constraints: Posting and propagation in the underlying temporal constraint network (time points and intervals)

More Flexible
Constraint Propagation

**Arc-Bounds**

Through Precedence Relations:

1. \( O_1 \rightarrow R_1 \) on [0,12] forward propagation
2. \( O_2 \rightarrow R_2 \) on [0,12] backward propagation
3. \( O_3 \rightarrow R_3 \) on [0,6] forward propagation
4. \( O_4 \rightarrow R_4 \) on [3,9] backward propagation

**Edge-Finding**

- \( S \) is a set of operations competing for resource \( R \).
- \( O \) is an operation not in \( S \) also requiring \( R \).

\[
\begin{align*}
(LFT(S) - EST(S) < Dur(O) + Dur(S)) & \quad \text{EST}(O) > \text{EST}(S) + Dur(S) \\
(LFT(S) - \text{EST}(O) < Dur(O) + \text{EST}(O)) & \quad \text{EST}(O) > \text{EST}(S) + Dur(S)
\end{align*}
\]

**Contention-based Ordering Heuristic**

- **Individual Demand of \( O_1 \) for \( R_j \)**
- **Individual Demand of \( O_2 \) for \( R_j \)**

**Contestion**: Aggregated curves found for each resource

**Critical Region**: Where a resource is contended the most

**Most Critical Unassigned Operation**: Contributes the largest area in critical region

**Variable Ordering Heuristic**: Choose the most critical unassigned operation
Slack-based Ordering Heuristic

(Precedence constraint-posting slack)

For two unordered operations I and J
- Slack(I \rightarrow J) = Lf_j - Est_i - (Dur_i + Dur_j)
- Bslack(I \rightarrow J) = Slack(I \rightarrow J) / f(S), (f(S) is similarity measure)

Min-Slack Selection (Variable Ordering)
- Choose operations pairs with minimum value of
  Min (Bslack(I \rightarrow J), Bslack(J \rightarrow I))

Max-Slack Posting (Value Ordering)
- Select the precedent constraint that leaves maximum remaining slack
  Max(Bslack(I \rightarrow J), Bslack(J \rightarrow I))

This slack-based heuristic performs competitively with contention-based heuristic
Significantly improved by combining with consistency enforcement methods (Baptiste, Le Pape, Nuijten, 1995)

Current State of Scheduling as CSP

Constraint-based scheduling techniques are an integral part of the scheduling suites by ILOG/I2
- Optimization results comparable to best approximation algorithms currently known on standard benchmark problems.
- Best known solutions to more idiosyncratic, “multi-product hoist scheduling” application (PCB electroplating).
- Better in most large-scale problems than IP formulations
Integrating Planning & Scheduling

Overview
- Why integrate planning and scheduling?
- Planning: The state of the art
- Scheduling: The state of the art
- Integrating Planning and Scheduling

Approaches
- Decoupled
  - Existing approaches
- Monolithic
  - Extend Planners to handle time and resources
  - Extend Schedulers to handle choice
- Loosely Coupled
  - Making planners and schedulers interact
Decoupled approaches
(which is how Project Mgmt Done now)

Management

Technology Development

Mid-lower manager
(task planning)

Implementers

MS Project
(scheduling)

Extending Planners

- ZENO [Penberthy & Weld], IxTET [Ghallab & Laborie], HSTS/RAX [Muscettola] extend a conjunctive plan-space planner with temporal and numeric constraint reasoners
- LPSAT [Wolfman & Weld] integrates a disjunctive state-space planner with an LP solver to support numeric quantities
- IPPlan [Kautz & Walser; 99] constructs ILP encodings with numeric constraints
- TGP [Smith & Weld; 99] supports actions with durations in Graphplan
Actions with Resources and Duration

Load(P:package, R:rocket, L:location)

Resources: ?h : robot hand
Preconditions: Position(?h,L) [?s, ?e]  
Free(?h) ?s  
Charge(?h) > 5 ?s

Effects:  
holding(?h, P) [?s, ?t1]  
depositing(?h,P,R) [?t2, ?e]  
Busy(?h) [?s, ?e]  
Free(?h) ?e  
Charge - .03*(?e - ?s) ?e

Constraints:  
?t1 < ?t2  
?e - ?s in [1.0, 2.0]

Capacity(robot) = 3

What planners are good for handling resources and time?

◇ State-space approaches have an edge in terms of ease of monitoring resource usage
  – *Time-point based representations are known to be better for multi-capacity resource constraints in scheduling*

◇ Plan-space approaches have an edge in terms of durative actions and continuous change
  – Notion of state not well defined in such cases (Too many states)
  – *PCP representations are known to be better for scheduling with single-capacity resources*
Extending Scheduling

Monolithic Architectures Scale Poorly

- Extended planning systems are hard to control
  - RAX uses a very error-prone hand-coded search control strategy
- Extended scheduling systems tend to lose effectiveness due to increased disjunction
- Monolithic systems can sometimes show counter-intuitive behavior (by multiplying search failures)
Loosely Coupled Architectures

Schedulers already routinely handle resources and metric/temporal constraints.
- Let the “planner” concentrate on causal reasoning
- Let the “scheduler” concentrate on resource allocation, sequencing and numeric constraints for the generated causal plan

Need better coupling to avoid inter-module thrashing....

Making Loose Coupling Work

- How can the Planner keep track of consistency?
  - Low level constraint propagation
    » Loose path consistency on TCSPs
    » Bounds on resource consumption,
    » LP relaxations of metric constraints
  - Pre-emptive conflict resolution
    The more aggressive you do this, the less need for a scheduler..
- How do the modules interact?
  - Failure explanations; Partial results
RealPlan--Master/Slave

(RealPlan-MS)

Master-Slave

P → S

Planner does causal reasoning.

Scheduler attempts resource allocation

If scheduler fails, planner has to restart

<table>
<thead>
<tr>
<th>Level</th>
<th>Action by level</th>
<th>Plan level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unitask R III, III</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Unitask R III, III</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Unitask R III, III</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Unitask R III, III</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Unitask R III, III</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>Pickup R IIIA</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>Stack R III, III</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>Stack R III, III</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>Stack R III, III</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>Stack R III, III</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2: shows policy and execution order of radiocells. is a unit of radio. is a time of radio.
Performance of Master-Slave Coupling

When scheduler fails, no specific guidance is given to the planner.

RealPlan: Peer-to-Peer

Explanation-directed backtracking between Planner and Scheduler.

Peer-Peer (RealPlan-PP)

Planner’s CSP:
Variables: “goals”
Values: “actions”

Scheduler’s CSP:
Variables: “Actions”
Values: “Resources”
Inter-module Dependency Directed Backtracking

Scheduler’s Task → Explanation Generation

Generate compact explanation of the Scheduler’s failure in allocating resources

Interface’s Task → Explanation Translation

Translate the explanation into a form that makes sense to the planner.

Planner’s Task → Generation of Alternative Plan

Use the translated explanation to generate a plan that avoids this failure.

Resource Domains:
A_1, A_2, A_3: \{R_1, R_2\}
A_4, A_5: \{S_1, S_2, S_3\}

Resource Constraints:
A_1 ≠ A_2; A_2 ≠ A_3;
A_1 ≠ A_4; A_3 ≠ A_4;

Subset of variables that cannot be assigned values (reason of failure):
(A_1, A_2, A_3)

Variable Assignments:
A_1 = R_1
A_2 = R_2
A_4 = S_1
A_3 = R_1

N_1: \{A_1 = R_1\}
N_2: \{A_1 = R_1, A_2 = R_2\}
N_3: \{A_1 = R_1, A_2 = R_2, A_4 = S_1\}
N_4: \{A_1 = R_1, A_2 = R_2, A_4 = S_1, A_3 = R_1\}
A temporal planner supporting causal plan synthesis

Next-Generation Realplan (NASA Sponsored)

Mission Profile
Mission modifications
Execution status
Replanning requests
Plans with annotated waypoints
Executor
CSP-based Finite capacity resource scheduler
Mixed Integer/Linear programming module for metric constraints
Summary & Conclusion

- Motivated the need for integrating Planning and Scheduling
- Discussed the state of the art in Planning and Scheduling
- Discussed approaches for integrating them
  - Loosely coupled architectures are a promising approach