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

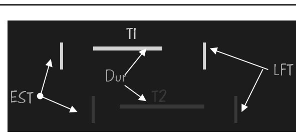
Find start times for jobs that are optimal (wrt make-spans, resource consumption etc)

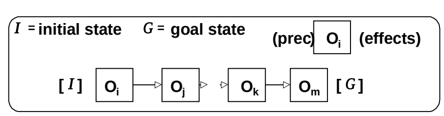
Resource Reasoning

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

Causal Reasoning





--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

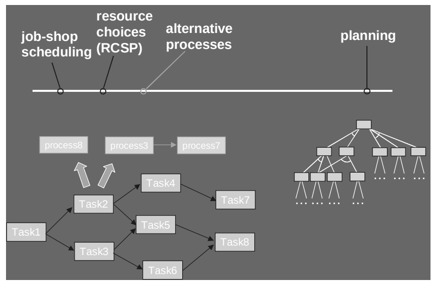
- Many real-world problems require *both* capabilities
  - Supply Chain Management problems
    - *I2, ILOG, Manugistics*
  - Planning in domains with durative actions, continuous change
    - NASA RAX experiment

Why now?

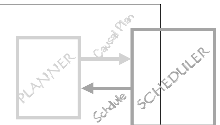
- Significant scale-up in plan synthesis in last 4-5 years
  - 5/6 action plans in minutes to 100 action plans in minutes
  - Breakthroughs in search space representation, heuristic and domain-specific
- Significant strides in our understanding of connections between planning and scheduling
  - Rich 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**



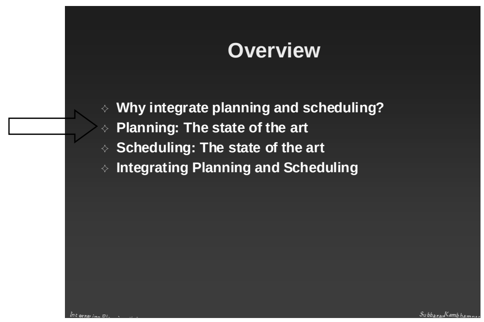
- ✧ **Coupled Architectures**
  - **De-coupled**
  - **Loosely Coupled (RealPlan System)**



Overview

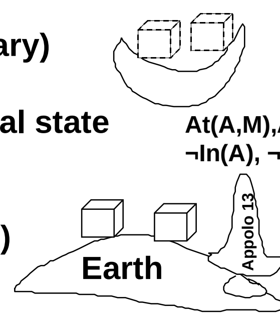
- ✔ Why integrate planning and scheduling?
- ➡ Planning: The state of the art
- ✧ Scheduling: The state of the art
- ✧ Integrating Planning and Scheduling

Planning:
The State of the Art



(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



At(A,M),At(B,M)
¬In(A), ¬In(B)

At(A,E), At(B,E),At(R,E)

Effects

In(o_1)

Load(o_1)

Preconditions

At(o_1,l_1), At(R,l_1)

¬In(o_1)

Unload(o_1)

In(o_1)

At(R,M), ¬At(R,E)

$\forall_x$ In (x) $\Rightarrow$ At (x, M) & ¬At(x, E)

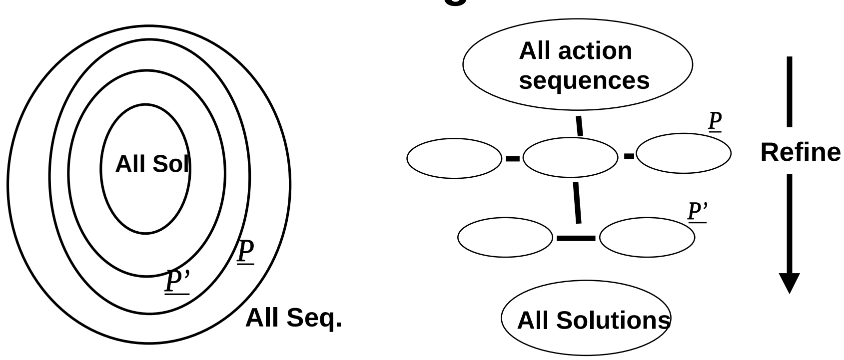
Fly()

At(R,E)

The (too) many brands of classical planners

- Planning as Theorem Proving (Green's planner) → Planning as Search
- Planning as Theorem Proving → Planning as Model Checking
- 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 States → Search in the space of Plans (total order, partial order, protections, MTC) (Interplan,SNLP,TOCL, UCPOP,TWEAK)
- Search in the space of States → Search in the space of Task networks (reduction of non-primitive tasks) (NOAH, NONLIN, O-Plan, SIPE)
- Search in the space of Plans → Search in the space of Task networks
- Search in the space of Plans - - → Planning as CSP/ILP/SAT/BDD
- Planning as Model Checking → Planning as CSP/ILP/SAT/BDD (Graphplan, IPP, STAN, SATPLAN, BLackBOX,GP-CSP,BDDPlan)

Refinement Planning: The idea



Refinements

Partial plans

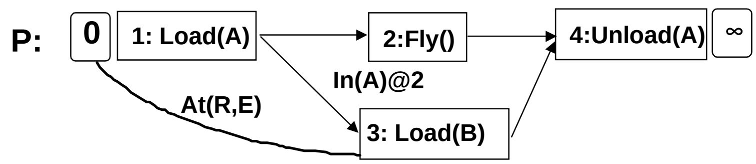
Narrowing *sets of action sequences* to progress towards sets of solutions

Remove non-solutions

[AIMAG-97]

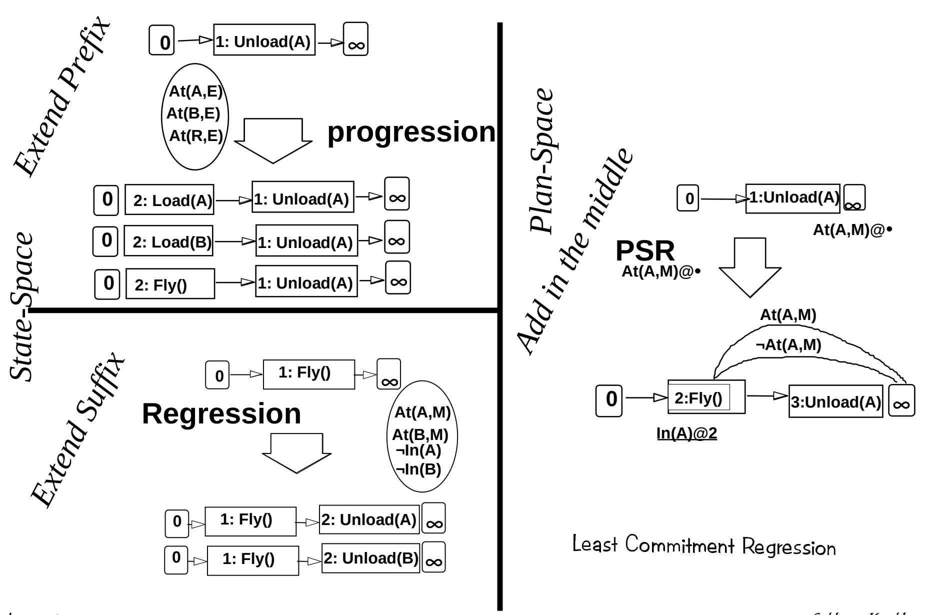
Plan Representation

Partial plan = ⟨Steps, Orderings, Aux. Constraints⟩



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

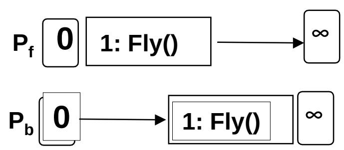


Tradeoffs among Refinements

The Tastes great/
Less Filling holy wars

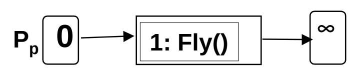
FSR and BSR must commit to both position and relevance of actions

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

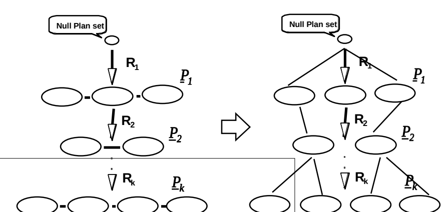


Plan-space refinement (PSR) avoids constraining position

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



A flexible Split & Prune search for Refinement Planning

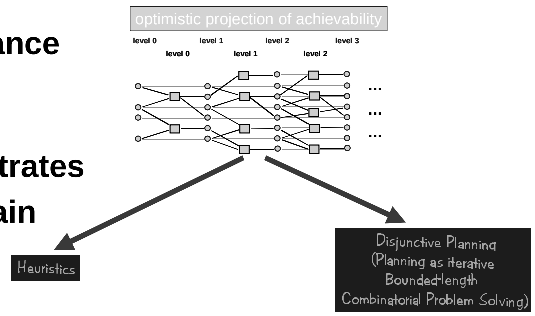


Refineplan(*P* : Plan)

0*. If «*P* » is empty, Fail.
1. If a minimal candidate of *P* is a solution, terminate.
2. Select a refinement strategy *R* . Appply *R* to *P* to get a new plan set *P'*
3. Split *P'* into *k* plansets
4. Non-deterministically select one of the plansets P'_i Call Refine(P'_i)

Broad Themes in the Planning Renaissance

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



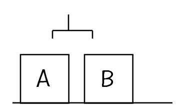
Tutorial on Recent Advances in AI Planning (IJCAI-99,AAAI-00)
http://rakaposhi.eas.asu.edu/planning-tutorial

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

Goal:
~clear(B), hand-empty

Pickup(A)

Pickup(B)



Progression Search

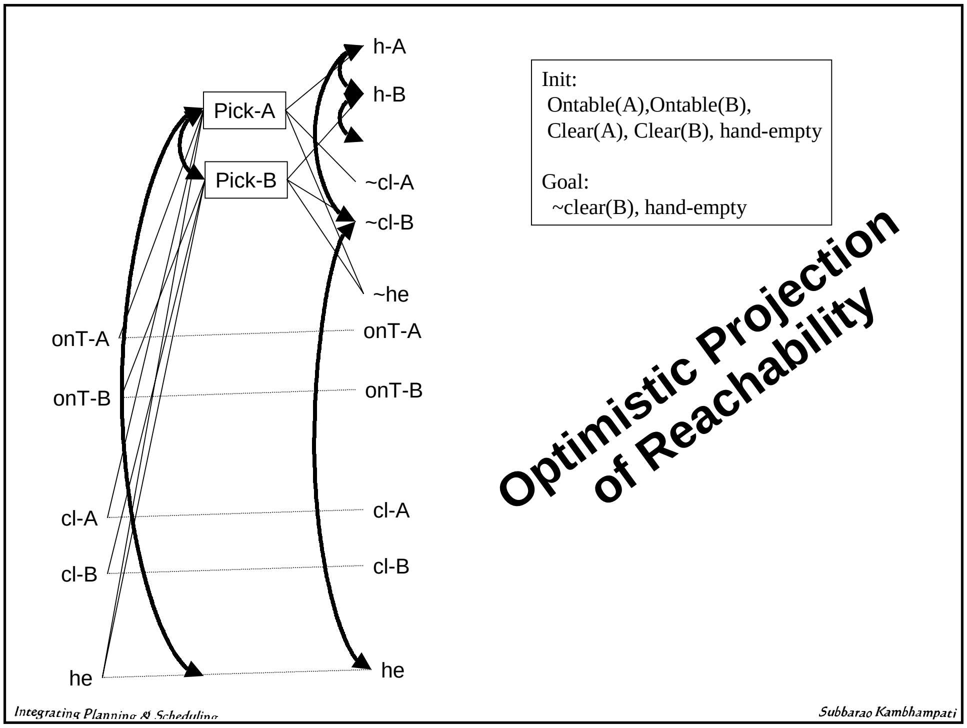
Regression Search

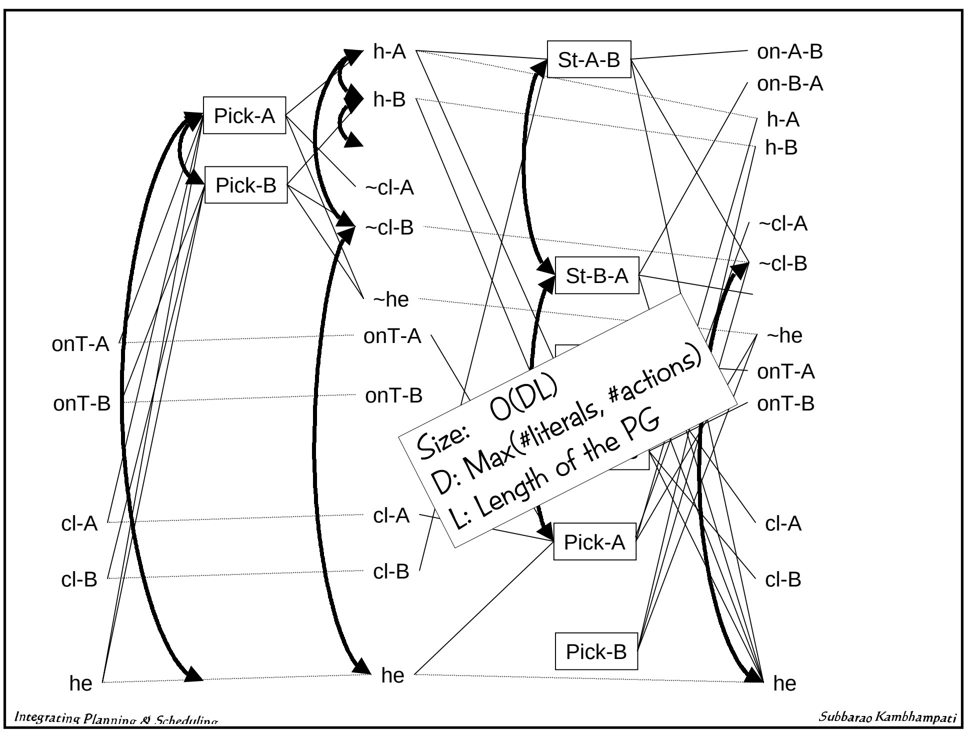
Putdown(A)

Stack(A,B)

Putdown(B)??

??How promising are the intermediate States/Plans?


Optimistic Projection of Reachability
Init:
Ontable(A),Ontable(B),
Clear(A), Clear(B), hand-empty
Goal:
~clear(B), hand-empty
Pick-A
Pick-B
h-A
h-B
~cl-A
~cl-B
~he
onT-A
onT-B
cl-A
cl-B
he
Integrating Planning & Scheduling
Subbarao Kambhampati


Size: O(DL)
D: Max(#literals, #actions)
L: Length of the PG
Pick-A
Pick-B
St-A-B
St-B-A
h-A
h-B
~cl-A
~cl-B
~he
onT-A
onT-B
cl-A
cl-B
he
on-A-B
on-B-A
Integrating Planning & Scheduling
Subbarao Kambhampati

Heuristics based on the Planning Graph

- 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 ∞
    - Else k+1 (k is the current length of the graph)

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

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

Yesterday's Talk (AltAlt System)

Adjusted Sum heuristic: [Sanchez *et. al., 2000*]

$H_{AdjSum2M}(S) = length(RelaxedPlan(S)) + \max_{p,q \in S} \delta(p,q)$ **Inadmissible**

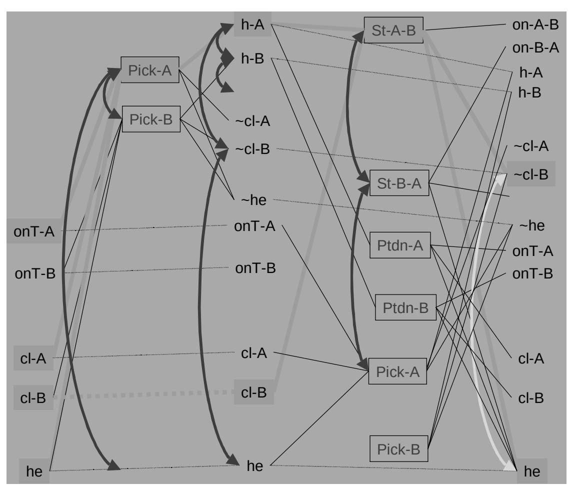
where $\delta(p,q) = lev(\{p,q\}) - \max\{lev(p), lev(q)\}$

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…)

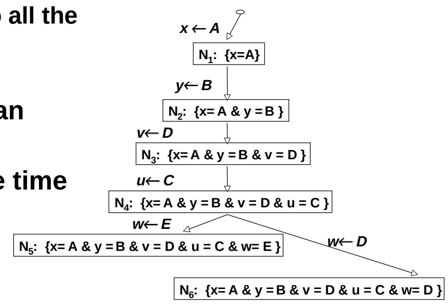


(very) Quick overview of CSP/SAT concepts

- **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**

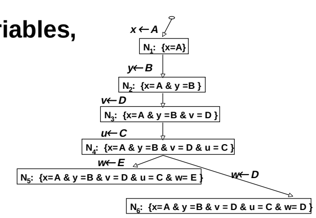
x,y,u,v: {A,B,C,D,E}
w: {D,E} l : {A,B}
x=A ⇒ w≠E
y=B ⇒ u≠D
u=C ⇒ l≠A
v=D ⇒ l≠B

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

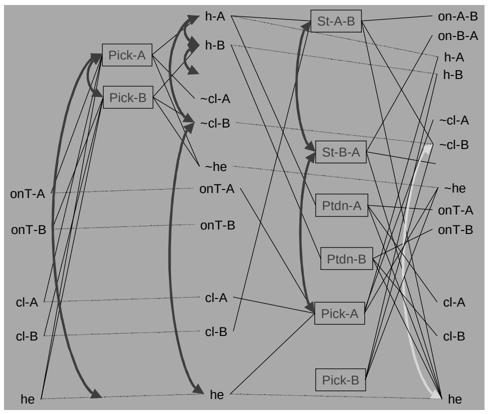
Consistency enforcement
k-consistency; adaptive consistency. (pre-processing)
Forward Checking, unit propagation during search (dynamic)

x,y,u,v: {A,B,C,D,E}
w: {D,E} l : {A,B}
x=A ⇒ w≠E
y=B ⇒ u≠D
u=C ⇒ l≠A
v=D ⇒ l≠B

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



Variables: Domains

~cl-B-2: { #, St-A-B-2, Pick-B-2}
he-2: {#, St-A-B-2, St-B-A-2, Ptdn-A-2,Ptdn-B-2}
h-A-1: {#, Pick-A-1}
h-B-1: {#,Pick-B-1 }
….

Constraints:
he-2 = St-A-B-2 => h-A-1 !=# {activation}
On-A-B-2 = St-A-B-2 => On-B-A-2 != St-B-A-2 {mutex constraints}
Goals:
~cl-B-2 != # he-2 !=#

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
-- 0/1 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 v ~Y v Z => x + (1 - y) + z >= 1**
  - **Explicitly set up tighter ILP inequalities**
    - **» If X,Y,Z are pairwise mutex, we can write x+y+z <= 1 (instead of x+y <=1 ; y+z <=1 ; z +x <= 1)**

[Walser & Kautz; Vossen et. al; Bockmayr & Dimopolous]

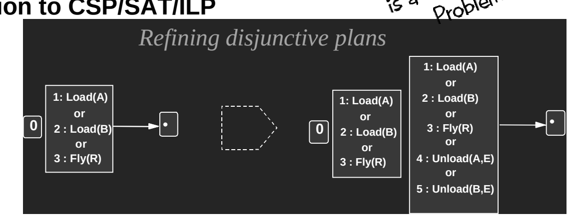
Relative Tradeoffs Offered by the various compilation substrates

- 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

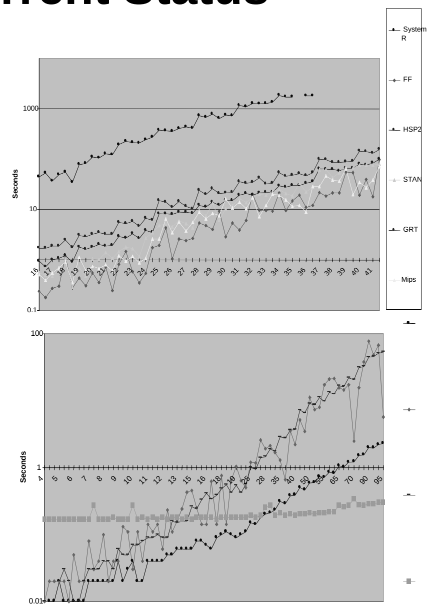
Solution Extraction is a combinatorial Problem



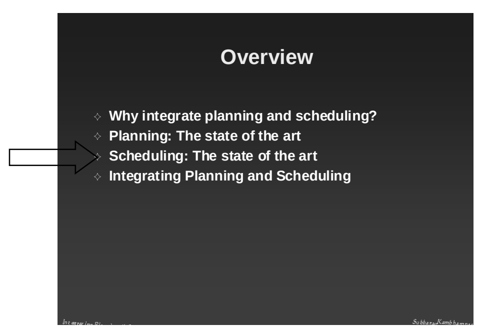
Planning: Current Status

- Disjunctive planners as well as heuristic state-search planners scale well
  - Plans with upto 100 actions synthesized in minutes
  - Impressive performances in AIPS-1998 and AIPS-2000 Competions
- Plan-space planners with comparable performance are still to be developed

  [Long & Kambhampati, IJCAI-2001 (may be)]

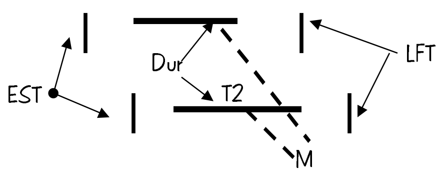


Scheduling: The State of the Art



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

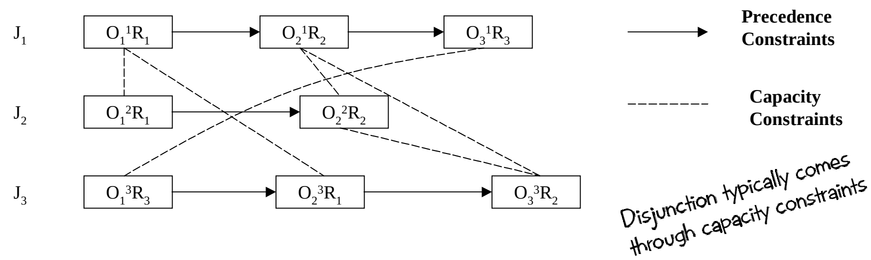


CSP Models
- Time point model
  - » Tasks as variables, Time points as values
  - » EST, LFT, Machine contention as constraints
- Inter-task precedences as variables (PCP model)

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



Start Point Representation

Variables: Start time st_i^l
Domain: $[est_i^l \ldots lst_i^l]$
Precedence constraints:
$$st_i^l + du_i^l \le st_j^l$$
Capacity Constraints:
$$st_i^l + du_i^l \le st_j^l \vee st_j^l + du_j^l \le st_i^l$$

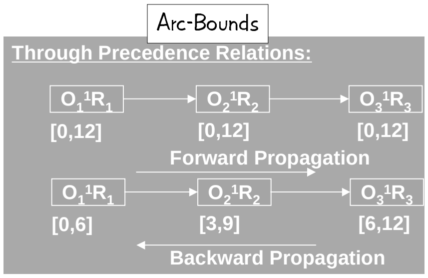
Simple Search

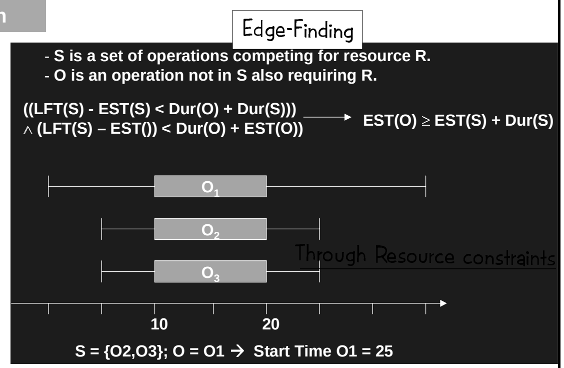
PCP Representation

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

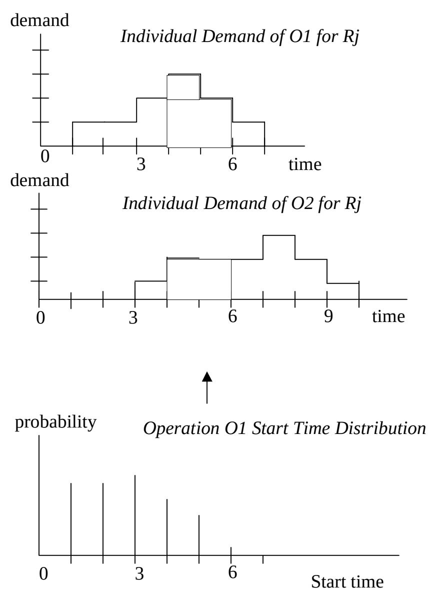
Constraint Propagation

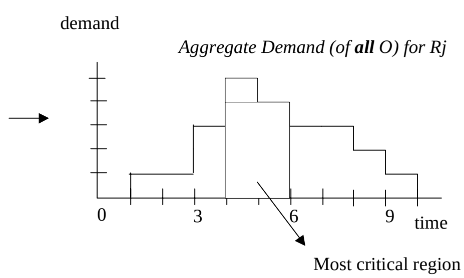




Contention-based Ordering Heuristic

[Sadeh, 1991]



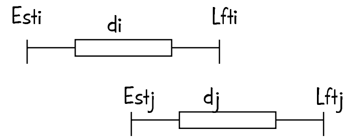


<u>Contention</u>: Aggregated curves found for each resource
<u>Critical Region:</u> Where a resource is contended the most
<u>Most Critical Unassigned Operation:</u> Contributes the largest area in critical region
<u>Variable Ordering Heuristic</u>: Choose the most critical unassigned operation

Slack-based Ordering Heuristic

(Precedence constraint-posting slack)

[Cheng&Smith, 1996]

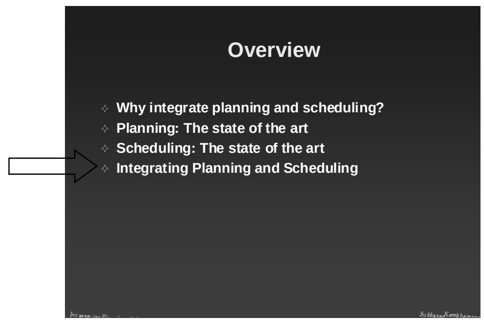


- **For two unordered operations I and J**
  - **Slack(I → J) = Lft_j – Est_i – (Dur_i + Dur_j)**
  - **Bslack(I → J) = Slack (I → J) / f(S),   (f(S) is similarity measure)**
- **Min-Slack Selection (Variable Ordering)**
  - **Choose opeations pairs with minimum value of**
    Min (Bslack(I → J), Bslack(J → I))
- **Max-Slack Posting (Value Ordering)**
  - **Select the precedent constraint that leaves maximum remaining slack**
    Max(Bslack(I → J), Bslack(J → 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**

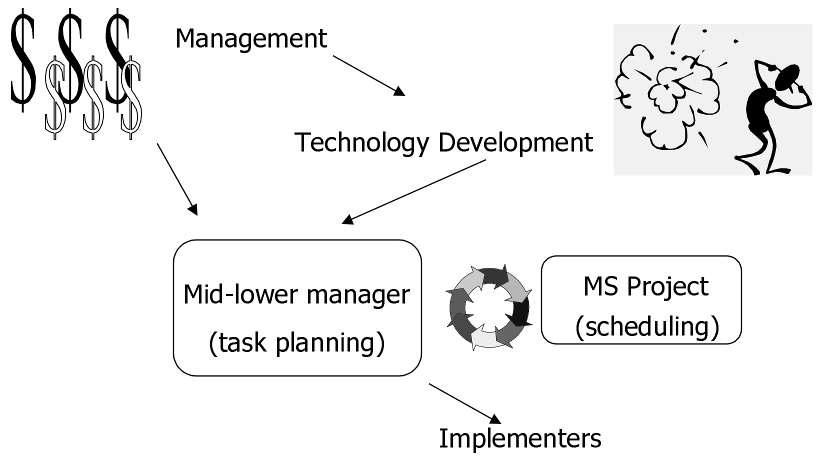
Integrating Planning & Scheduling



Approaches

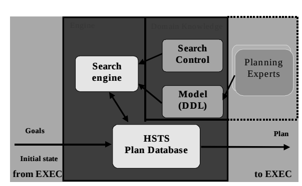
- **Decoupled**
  - Existing approaches
- **Monolithic**
  - Extend Planners to handle time and resources
  - Extend Schedulers to handle choice
- **Loosely Coupled**
  - Making planners and schedulers interact

Decoupled approaches (which is how Project Mgmt Done now)



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

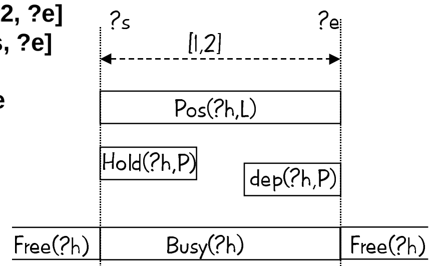
Load(P:package, R:rocket, L:location)

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]

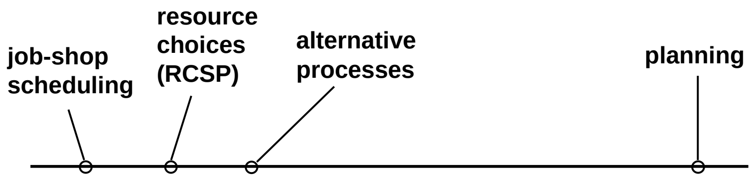


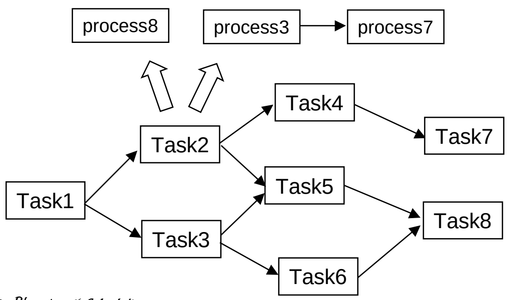
Capacity(robot) = 3

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



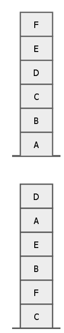


Ordering choices
Resource choices
Process choices

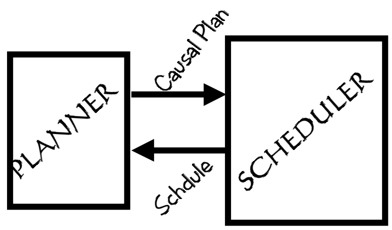
Monolithic Architectures Scale Poorly

- 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)

Performance worsens with increased resources!



Loosely Coupled Architectures



Schedulers already routinely handle resources and metric/temporal constraints.

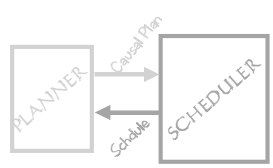
- Let the "planner"concentrate on causal reasoning
- Let the "scheduler" concentrate on resource allocation, sequencing and numeric constraints for the generated causal plan

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

[Srivastava & Kambhampati ECP 99; AAAI, 2000]

(RealPlan-MS)

Planner does causal reasoning.

Scheduler attempts resource allocation

If scheduler fails, planner has to restart

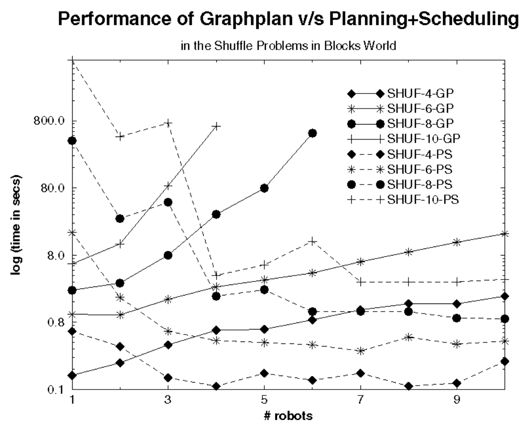
Level	Actions by level	# Robots
1	Unstack_R_blkF_blkE	1
2	Unstack_R_blkE_blkD	2
3	Unstack_R_blkD_blkC	3
4	Unstack_R_blkC_blkB	4
5	Putdown_R_blkC	
5	Unstack_R_blkB_blkA	5
6	Stack_R_blkF_blkC	5
6	Pickup_R_blkA	
7	Stack_R_blkB_blkF	4
8	Stack_R_blkE_blkB	3
9	Stack_R_blkA_blkE	2
10	Stack_R_blkD_blkA	1

Allocation Policy	Constraint on values
Maintain concurrency (Class INFRES)	$PA_i = i$, $PA_j = j$, $RA_i = RA_j = \{1,..N\}$ $PF_{ij} = PU_{ij} =$ $RF_{ij} = RU_{ij} = \perp$
Serialize plan	$PA_i = \{i,..L^{MAX}\text{-}1\}$, $PA_j = \{j,..L^{MAX}\}$, $RA_i = RA_j = \{1,..N\}$ $PF_{ij} = PU_{ij} =$ $RF_{ij} = RU_{ij} = \perp$
Introduce Free/ Reallocate action	(Class FINRES)
Class FIX	$PA_i = i$, $PA_j = j$, $RA_i = RA_j = \{1,..N\}$ $PF_{ij} = \{\perp, i+1\}$, $PU_{ij} = \{\perp, j\text{-}1\}$, $RF_{ij} = RU_{ij} = \{\perp, 1,..N\}$
Class SAMELEN	$PA_i = \{i,..L\text{-}1\}$, $PA_j = \{j,..L\}$, $RA_i = RA_j = \{1,..N\}$ $PF_{ij} = \{\perp, i+1,..L\text{-}2\}$, $PU_{ij} = \{\perp, j\text{-}1,..L\text{-}1\}$, $RF_{ij} = RU_{ij} = \{\perp, 1,..N\}$
Class INCRLEN	$PA_i = \{i,..L^{MAX}\text{-}1\}$, $PA_j = \{j,..L^{MAX}\}$, $RA_i = RA_j = \{1,..N\}$ $PF_{ij} = \{\perp, i+1,..L^{MAX}\text{-}2\}$, $PU_{ij} = \{\perp, j\text{-}1,..L^{MAX}\text{-}1\}$, $RF_{ij} = RU_{ij} = \{\perp, 1,..N\}$

Table 1: *Allocation policy and restrictions on values of variables. L^{MAX} is some maximum length ($L^{MAX} > L$) upto which the steps of the plan can be increased.*

Performance of Master-Slave Coupling

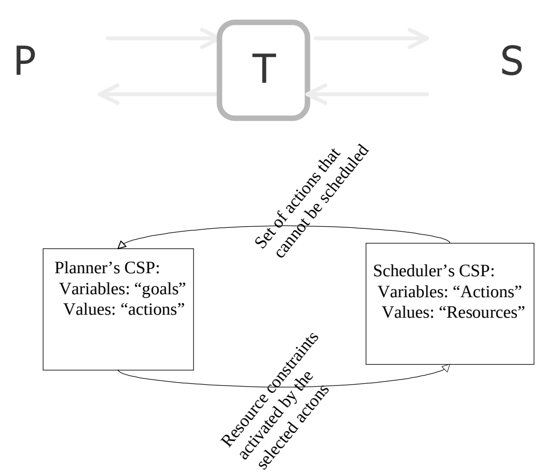
When scheduler fails, no specific guidance is given to the planner



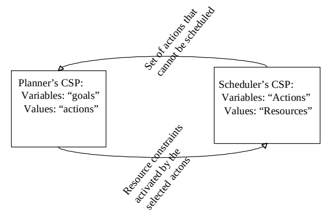
RealPlan: Peer-to-Peer

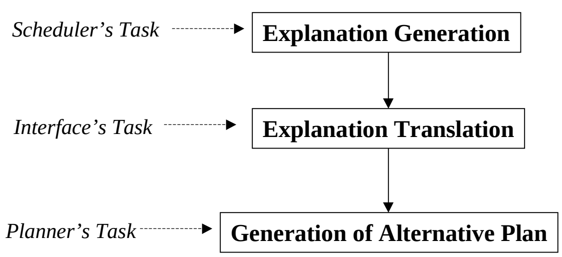
Explanation-directed backtracking between Planner and Scheduler

Peer-Peer (RealPlan-PP)



Inter-module Dependency Directed Backtracking





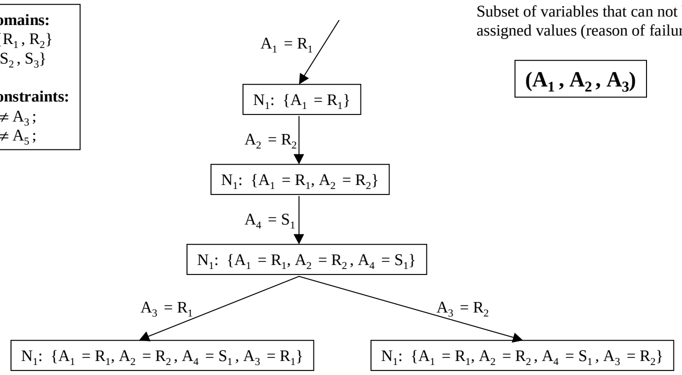
Generate compact explanation of the Scheduler's failure in allocating resources

Translate the explanation into the form that make sense to the planner.

Use the translated explanation to generate plan that avoid this failure.

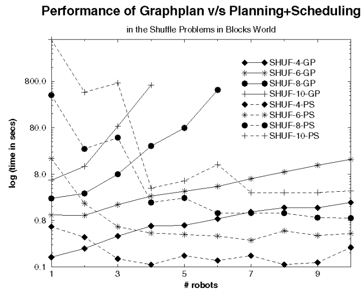
Resource Domains:
A_1, A_2, A_3: $\{R_1, R_2\}$
A_4, A_5: $\{S_1, S_2, S_3\}$

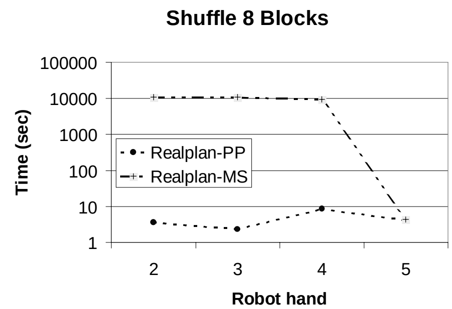
Resource Constraints:
$A_1 \neq A_2$; $A_2 \neq A_3$;
$A_1 \neq A_3$; $A_4 \neq A_5$;



Subset of variables that can not be assigned values (reason of failure):

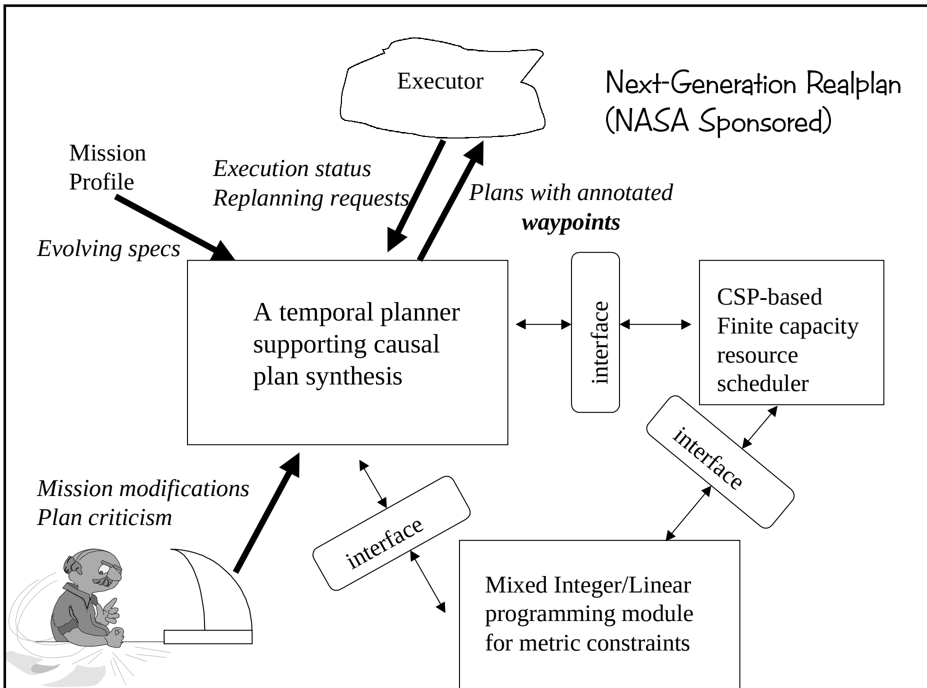
(A_1, A_2, A_3)





Problem	RealPlan-PP Class	RealPlan-PP Time	RealPlan-PP #bk	BTPWF Time	RealPlan-MS Class	RealPlan-MS Time	Speedup (x) BTPWF	Speedup (x) RealPlan-MS
Shuf_b8r4	INFRES	2.52	4	>3 hr	SAMELEN	9373	**>4170**	**3619**
Shuf_b8r4	FIX	8.74	1	>3 hr	SAMELEN	9373	**>1236**	**1072**
Huge_r6	***INFRES***	***4.53***	***4***	***976***	***INCRLEN***	***11056***	***215***	***2441***
Huge_r6	FIX	17.48	3	991	INCRLEN	11056	**56.7**	**632**
Huge_r6	SAMELEN	20.59	2	983	INCRLEN	11056	**47.7**	**537**

Next-Generation Realplan (NASA Sponsored)



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