Integrating Planning, Execution, and Monitoring (IPEM)

Abrose-Ingerson and Sam Steel
IPEM Framework

- Set of plan transformations which remove flaws for the current plan

- at each point in execution try to transform the current plan into a viable one (e.g., one that applies in the current situation to achieve desired goals)
Assumptions

- Current world description (CWD)
  - can change
    - actions can fail to achieve purported effects
    - actions can have delayed effects (but told when actions over “time out”)
  - CWD may be missing true facts (e.g., sound but not complete)
Chapman Terminology

- Current plan is incomplete, partial
  - partial commitment on codesignation (assignment of variables to constants)
  - partial commitment on ordering of actions (precedence relations may not cause total order)
  - completion of a partial plan $P$ satisfies every constraint in $P$

- Partial plan $P$ *necessarily* satisfies $S$ if $S$ holds in every completion

- $P$ *possibly* satisfies $S$ if $S$ holds in at least one completion
A Plan is...

- **BEGIN action**
  - postconditions are CWD

- **END action**
  - preconditions are desired goals

- **set of protections (ranges, causal links)**
  - connects postcondition (supplier) with precondition (user)
    - postcondition and precondition must codesignate (unify)
    - supplier necessarily before user

- **set of actions (suppliers and users)**
Sample Problem

... clear C
clear A
A on B

Begin

A on C

End

A
B
C
D

t.1
t.2
t.3
Plan Generation

- Also used in modification (incremental modifications on a partial plan)
  - unsupported precondition
  - unresolved conflict
  - unexpanded action
Plan Modifications - Reduce

- Unsupported precondition; action A has precondition P with no range) (i.e., no supplier yet)

- use an action S already in plan such that S *possibly* before A and with effect E that *possibly* codesignates with P
  - make S necc. before A
  - make E necc. codes. with P

- add action to plan that satisfies above
Sample Plan

... clear C
clear A
A on B

Begin

~clear z
clear x
x on y

Move A from B to C

~clear C
clear A
A on B

Unsupported precondition
use exist. act.

~clear z
clear y
~(x on y)
x on z

A on C

End

~clear C
clear B
~(A on B)
A on C

Unsupported precondition
add action
Unresolved Conflict - Linearize

- If
  - plan has range R that protects P such that
  - there is an action A and
    - A is possibly after producer of R
    - A is possibly before consumer of R
    - A has an effect which asserts \( \neg P \) (possibly codes with \( \neg P \))
  - resolve by
    - A necessarily after consumer of R; or
    - A necessarily before producer of R
Example Conflict Resolution

Act 1:
Move D from C to E

Act 2:
Move A from B to C

Act 3:
Move C from F to H

Protect clear C from Act 1 to Act 3

Move act 2 before act 1

clear C

~clear C

clear B

~(A on B)

A on C

move act 2 after act 3
Unexpanded Action - Expand

- A plan has an action A which is not primitive
  - resolve by replacing action with its expansion
    (network of activities)
Plan Monitoring & Execution

- unsupported range
- unexecuted action
- timed out action
- unextended range
- redundant action
Unsupported Range

- Plan has a range (causal link) R which protects a proposition not in CWD
- fixed by removing the range
  - immediately creates an unsupported precondition on consumer (R)
  - resolve as in plan generation
Sample Plans

... clear C clear A A on B
Begin

clear z clear x x on y
Move A from B to C

~clear z clear y ~(x on y) x on z

~clear C clear B ~(A on B) A on C
End

Suppose we execute this plan but then somehow mistakenly A gets moved from B to D
Sample Plans

Begin

... clear C
clear A
A on B

Clear

clear z
clear x
x on y

Move A from B to D!

~clear z
clear y
~(x on y)
x on z

A on C

End

~clear D
clear B
~(A on B)
A on D

Unsupported range
remove it!
**Unexecuted Action: Execute**

- Action A ready for execution if:
  - A is primitive and is not END
  - all preconditions supported and produced by BEGIN
  - it is immediately after an executed action
  - it is not involved in an unresolved conflict
  - there is no “live” action B (i.e., not timed out) before A that expects a postcondition that can clobber (threat) any of A’s postconditions
Executing an Action A

- Make all parallel actions necessarily after A
  - what about currently executing actions?
- Calling associated procedure
  - (actual work)
- list postconditions as expected (monitoring)
- recording action as executed
Unextended Range: Extend

- Can remove a range $R$ and replace it with $R'$ where
  - $R$ and $R'$ have the same consumer
  - producer of $R'$ is necessarily before producer of $R$
  - $R'$ does not create an unresolved conflict flaw

- special case when activity $A$ executes, achieves desired effect $E$, range moves from producer=$A$ to producer=$BEGIN$

- seems like an optimization
Timed Out Action: Excise Action

- When an action is terminated and no further effects occur
- Presumes that such a time can be specified and that actions can be terminated
- Remove action and all its ranges from plan
Redundant action: Excise Action

- An action not producing for any range has a redundant action flaw and is excised
- Excising a redundant action can result in further action redundancies
Control: The Scheduler

- IPEM maintains an agenda of tasks
  - each agenda item is a flaw
  - with the possible fixes
- IPEM orders flaws

Flaw ordering heuristic
- unsupported range
- unextended range
- timed-out action
- unresolved conflict
- unsupported precondition
- unexpanded action
- unexecuted action

Within Flaw Class
- select the flaw with fewest options
  (corresponds to CSP value ordering heuristic)
Example: Replanning

Initial plan
...
clear C
clear A
A on B

Begin

Move A
from B to C

Move u.2
from t.5
to r.1

End

A on C

~clear z
clear y
~(x on y)
x on z

u.m on r.n

m=2	n=1

x=u.2
y=t.5
z=r.1

Initial plan
...
clear C
clear A
A on B

Begin

~clear z
clear y
~(x on y)
x on z

u.m on r.n

m=2	n=1

x=u.2
y=t.5
z=r.1

u.2 on t.5
clear u.2
clear r.1
clear r.2

x on y
clear x
~clear z

x on z
~(x on y)
clear y
~clear z

A on C

~clear z
clear y
~(x on y)
x on z
Example: Replanning

Baby moves d from t.3 to r.1

... clear C clear A A on B

Begin

u.2 on t.5 clear u.2 clear r.1 clear r.2...

Move A from B to C

Move u.2 from t.5 to r.1

~clear z clear x x on y

A on C

~(x on y) x on z

End

u.m on r.n

m=2

n=1

x=u.2

y=t.5

z=r.1

x=A

y=B

z=C

C B A D
t.1 t.2 t.3 t.4 t.5 t.6

x=A

y=B

z=C

m=2

n=1

x=u.2

y=t.5

z=r.1

Baby moves d from t.3 to r.1

... clear C clear A A on B

Begin

u.2 on t.5 clear u.2 clear r.1 clear r.2...

Move A from B to C

Move u.2 from t.5 to r.1

~clear z clear x x on y

A on C

~(x on y) x on z

End

u.m on r.n

m=2

n=1

x=u.2

y=t.5

z=r.1

x=A

y=B

z=C

C B A D
t.1 t.2 t.3 t.4 t.5 t.6

x=A

y=B

z=C

m=2

n=1

x=u.2

y=t.5

z=r.1

Baby moves d from t.3 to r.1

... clear C clear A A on B

Begin

u.2 on t.5 clear u.2 clear r.1 clear r.2...

Move A from B to C

Move u.2 from t.5 to r.1

~clear z clear x x on y

A on C

~(x on y) x on z

End

u.m on r.n

m=2

n=1

x=u.2

y=t.5

z=r.1

x=A

y=B

z=C

C B A D
t.1 t.2 t.3 t.4 t.5 t.6

x=A

y=B

z=C

m=2

n=1

x=u.2

y=t.5

z=r.1

Baby moves d from t.3 to r.1

... clear C clear A A on B

Begin

u.2 on t.5 clear u.2 clear r.1 clear r.2...

Move A from B to C

Move u.2 from t.5 to r.1

~clear z clear x x on y

A on C

~(x on y) x on z

End

u.m on r.n

m=2

n=1

x=u.2

y=t.5

z=r.1

x=A

y=B

z=C

C B A D
t.1 t.2 t.3 t.4 t.5 t.6
Example: Replanning

Begin

Move1 A from B to C

Move2 u.2 from t.5 to r.2

~clear z
clear x
x on y
x on z

~(x on y)

A on C

End

u.m on r.n

x=u.2
y=t.5
z=r.2

m=n=2

x=A
y=B
z=C

Reassign end. n=2
Move2. z=r.2
...
clear C
clear A
A on B
Example: Replanning

Baby moves A from B to u.2
- clear B
- clear C
- clear A
- A on B
- A on u.2

Begin

<table>
<thead>
<tr>
<th>t.1</th>
<th>t.2</th>
<th>t.3</th>
<th>t.4</th>
<th>t.5</th>
<th>t.6</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Move1 A from B to C
- A on C
- u.m on r.n
- m=2
- n=2
- x=u.2
- y=t.5
- z=r.2

Move2 u.2 from t.5 to r.2
- x on y
- ~clear z
- A on u.2

x = A
y = B
z = C

Clearations:
- clear z
- clear x
- x on y
- ~clear z
- clear y
- ~(x on y)
- x on z
- x on y
- ~clear z
- A on B
- A on u.2
- u.2 on t.5
- clear u.2
- clear r.1
- clear r.2
- D
- r.1
- A
- u.1
- u.2
- r.2
Example: Replanning

Fix lower flaw

clear B
clear C
clear A
A on B
A on u.2

Begin

Move1 A from B to C

Move2 u.2 from t.5 to r.2

End

u.m on r.n

x=A
y=B
z=C

m=2
n=2
x=B
y=t.5
z=r.2
Example: Replanning

Begin

clear B
clear C
clear A
A on B
u.2 on t.5
clear u.2
clear r.1
clear r.2
...
D
r.1
A
u.1
u.2
r.2

Move1 A from B to C

clear z
clear x
x on y

Move2 u.2 from t.5 to r.2

x on z

~clear z

End

x=A
y=u.2
z=C
x on z

~clear z

~(x on y)
x on z

x on y

~(x on y)
x on z

~clear z

x on y

~clear z

x=B
y=t.5
z=r.2

u.m on r.n
m=2
n=2

Fix upper flaw

x=A
y=u.2
z=C
x=B
y=t.5
z=r.2
Interleaving Planning and Execution

BEGIN

ASK

GET meal

DINE

END

... person likes thing guest likes meal

... have meal have meal dinner with guest dinner with Sandy
Interleaving Planning and Execution

Begin

ASK

GET meal

DINE

END

Sandy likes fondue

person likes thing

guest likes meal

have meal

have meal

dinner with guest

dinner with Sandy
Discussion

- Enables replanning for “classical planning”
- atomic actions (but parallel execution)
- does not enable reasoning about resources, metric time