Constraint Programming Scheduling Applications

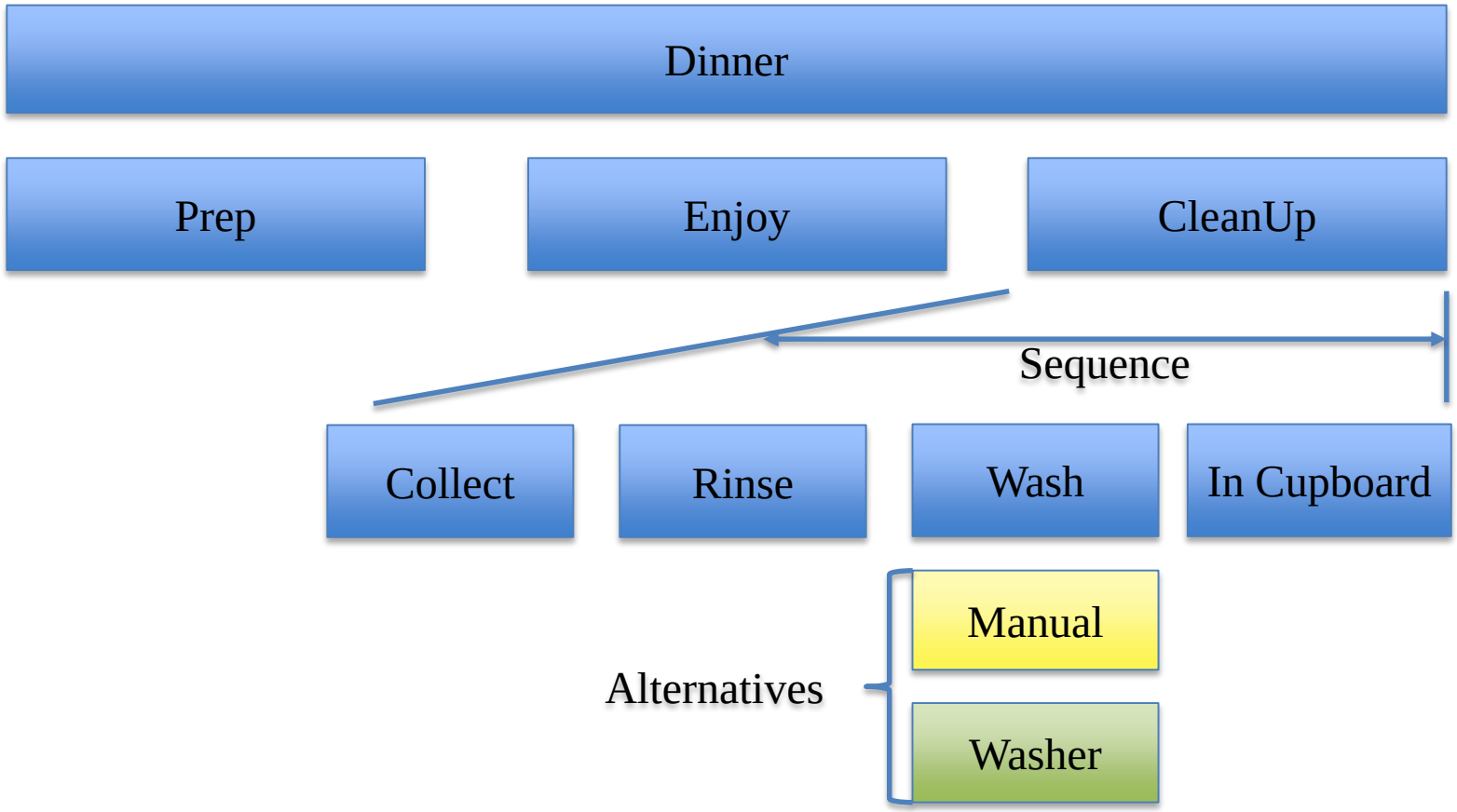
Chris Kuip
September 16, 2015

Contents

- Introduction to Scheduling with Constraint Programming
- Maintenance scheduling
- Bridge building

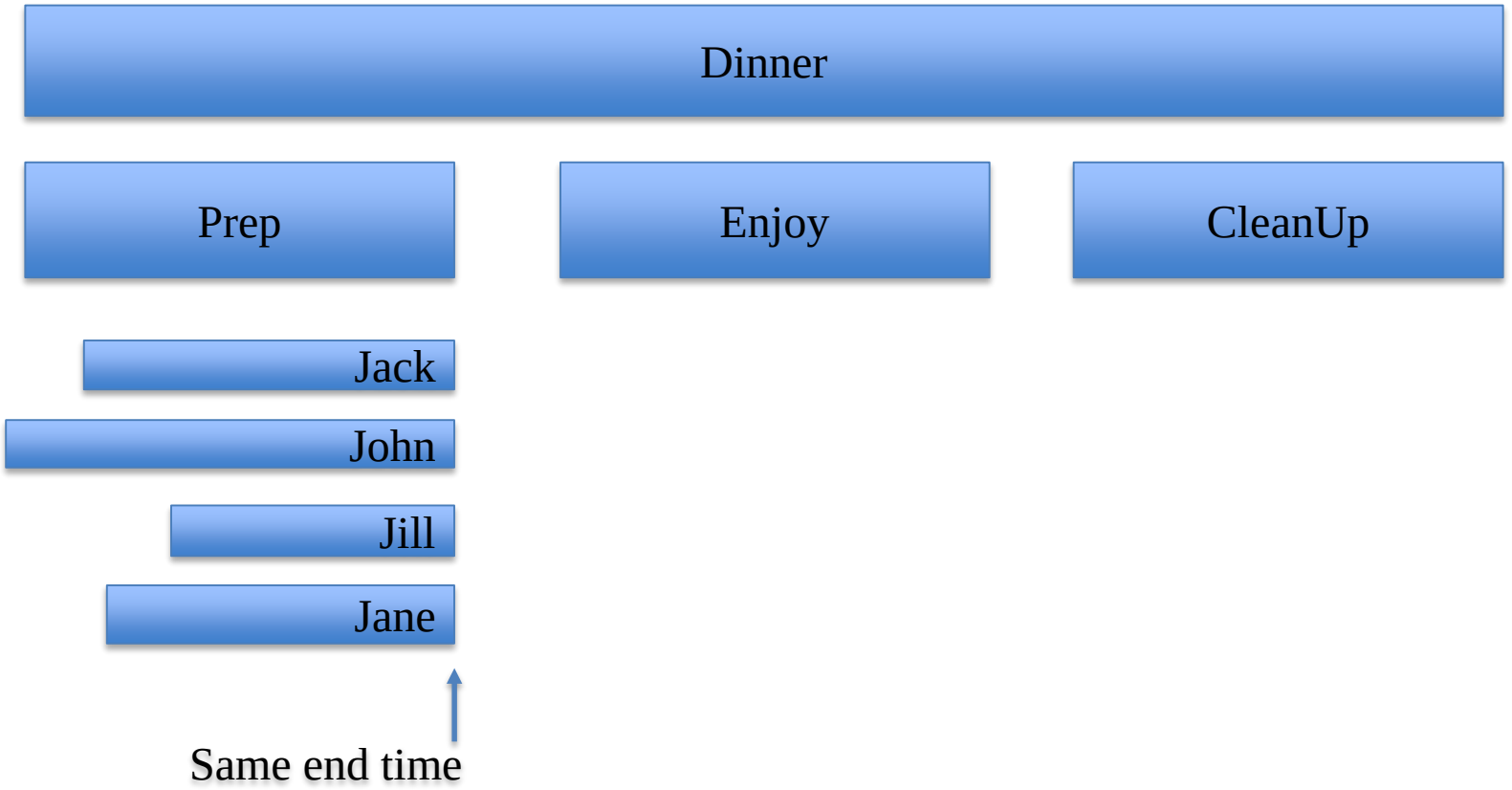
AIMMS

Trivial example: Dinner – Looking to CleanUp



AIMMS

Trivial example: Dinner – Looking at preparation



IBM/ILOG CP Optimizer technology used

- Special modeling constructs for scheduling
- Many years of research on algorithm development

AIMMS

Overview AIMMS scheduling support
1. Mathematical Program

We start with establishing a timeline (scheduleDomain):

- Contiguous subset of Integers
- Calendar, or a contiguous subset thereof

Specified in the Mathematical Program, example:

```
MathematicalProgram findSchedule {
        ...
        ScheduleDomain: Ca_Quarters;
}
```

AIMMS

Overview AIMMS scheduling support
2. Activities A. Declaration.

Declaring activities:

1. Which activities are there?
- indexDomain: (i,j,…) | condition(i,j,…)
- Property: Optional

2. What do we know about these activities?
- scheduleDomain attribute, for instance:
- A. timeline
- B. { startTimeWindow .. endTimeWindow }
- Length(and perhaps size), defining constraint, for instance:
- A. 5
- B. 3 * requiredNumberUnitsCreated

AIMMS

Overview AIMMS scheduling support
2. Activities B. Suffices and simple constraints

An activity A consists of 5 integer variables represented by suffices:

- A.begin
- A.end
- A.length and A.size
- A.present

These suffices can be used to set up simple constraints such as:
DinnerPrep.End <= DinnerEnjoy.begin ! Cook before enjoy.
DinnerPrep.End = DinnerPrepPerson(p).end ! Dinner is ready!

An activity A is active from A.begin
up to but NOT including A.end
Notation: [A.begin, A.end)

Constraint that is implicitly enforced on every activity A:

```
If A.Present Then
    A.end = A.Begin + A.Length
Endif
```

AIMMS

Overview AIMMS scheduling support
3. Precedence constraints

Relate two activities on .begin, .end and using '=' or '<='

```
CP::EndBeforeBegin(prep,enjoy);
```

Equivalent to:

```
If prep.present and enjoy.present then
    prep.end <= enjoy.begin
Endif ;
```

The filtering for these precedence constraints is highly effective.

CP::BeginAtBegin(a,b,d)
CP::BeginAtEnd(a,b,d)
CP::EndAtBegin(a,b,d)
CP::EndAtEnd(a,b,d)
CP::BeginBeforeBegin(a,b,d)
CP::BeginBeforeEnd(a,b,d)
CP::EndBeforeBegin(a,b,d)
CP::EndBeforeEnd(a,b,d)

Only constrain if activities a and b are both present.

AIMMS

Overview AIMMS scheduling support
4. Scheduling constraints

- An activity consists of several sub-activities

cp::Span(g, i, a(i))

- There are several alternative ways to realize an activity

cp::Alternative(g, i, a(i))

- Several activities need to take place at the same time:

cp::Synchronize(g, i, a(i))

Only constrain if activities g and a(i) are present.

AIMMS

Overview AIMMS scheduling support
5. Sequential resources

Resources that handle only one activity at a time.

Mandatory Attributes:

- scheduleDomain: timeline
- Activities: ajob, bjob(i), cjob(i,j)|allowedJob(i,j)

Optional Attributes:

- indexDomain: (i,j,...) | condition(i,j,…)

Optional attributes particular to sequential resources

- Transition: (job(i),job(j)) : times(i,j)
- firstActivity: ajob
- lastActivity: bjob('John')
- comesBefore: (ajob, bjob) ! Directly before
- precedes: (ajob,bjob) ! Somewhere before

AIMMS

Demo Maintenance Scheduling

This app:

- One maintenance group available for maintenance
- Large tasks - large time window
- Small tasks - small time window

AIMMS

Overview AIMMS scheduling support
6. Parallel resources a. Declaration

Resources that handle multiple activities at a time.

Mandatory Attributes:

- scheduleDomain: timeline
- Activities: ajob, bjob(i), cjob(i,j)|allowedJob(i,j)

Optional Attributes:

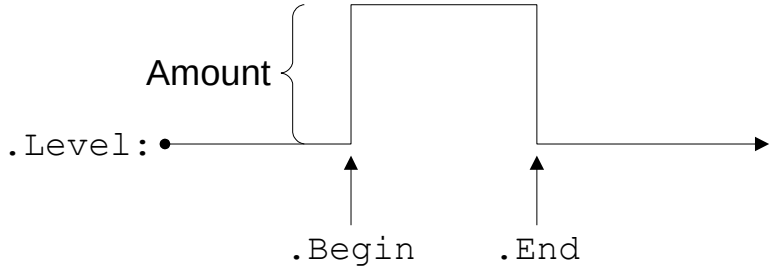
- indexDomain: (i,j,...) | condition(i,j,…)

Attributes particular to parallel resources

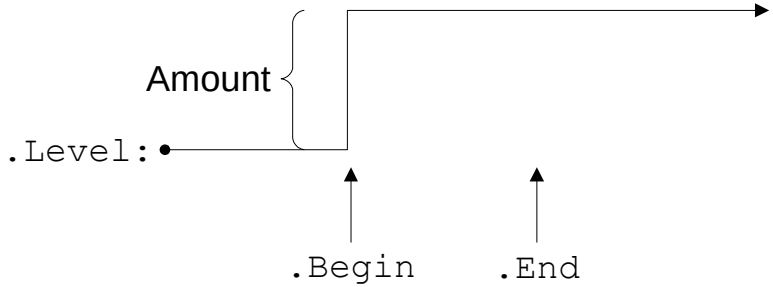
- levelRange: {0..pNumberOfShips}
- initialLevel: ajob
- levelChange: ajob : 1
- beginChange: bjob(i) : 1
- endChange: cjob(i,j): 2

AIMMS

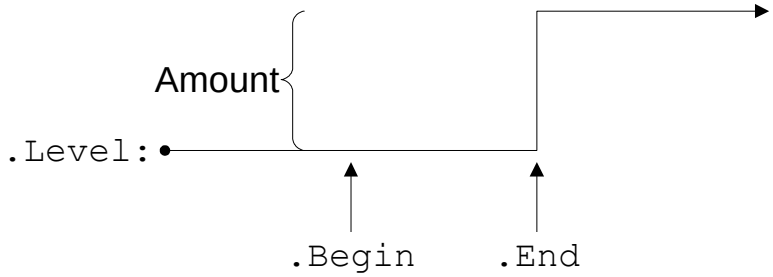
Overview AIMMS scheduling support
6. Parallel resources b. Pulses



Level change
or pulse



Begin change



End change

AIMMS

Overview AIMMS scheduling support
7. Difference .length and .size

Activities and resources may not be available every time slot.

For instance, human workers prefer to have weekends off.

Length does not suffice to measure amount of work done.

An activity A is active on timeslot t if
    t is in scheduleDomain of A
And
    t is in scheduleDomain of resource R,
        for every R on which A is scheduled.

For every activity A, we define A.size as
the number of active timeslots in the range [A.begin,A.end)

Implicit constraint:
If a.present then
      a.size <= a.length
Endif

AIMMS

Demo Bridge Building

Build a bridge

- several components to be placed,
- several tools to be used.

Interesting:

- Activity
- Constraint
- Parallel Resource
- Gantt Chart

AIMMS

Next webinar

- ?.
- ?
- October 2015

AIMMS

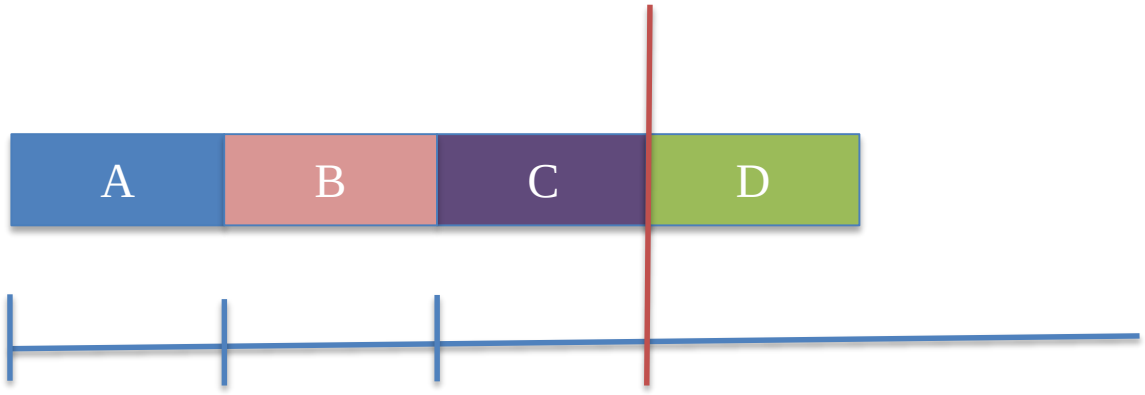
Questions?

AIMMS

Scheduling example

4 non overlapping activities A, B, C, D; each length 1, D comes last

Precedes(A, D)
Precedes(B, D)
Precedes(C, D)
NoOverlap(A,B,C,D)



From the above, scheduling engines can conclude begin of D >= 3.

AIMMS

Introduction to sequential resources

- Purpose: Scheduling sequences of activities
- Example:

```
Resource res {
    Usage: sequential;
    ScheduleDomain: TimeLine;
    Activities: Act(j);
    Transition: (act(j), act(k)) : ChgOver(j, k);
}
```

AIMMS