Constraint Programming for solving scheduling problems

FEUP / MIEIC
MPES 2010

João Gradim
Mário Carneiro

Summary

- What is constraint programming?
- Why CP for solving scheduling problems?
- How can we use CP to solve scheduling problems?

What is constraint programming?

- Programming paradigm
- Relations between variables are expressed as constraints
- We specify the properties of the desired solution instead of the steps we would take to find it
  - Compare with imperative programming

A simple example

- Find values for each of the letters in the puzzle
  - Digits are represented by letters
  - A possible formulation would be:
    - S ≠ E ≠ N ≠ D ≠ M ≠ O ≠ R ≠ Y
    - 1000**S** + 100**E** + 10**N** + **D** +
1000**M** + 100**O** + 10**R** + **E** =
10000**M** + 1000**O** + 100**N** + 10**E** + **Y**
  - *Variables and constraints*
  - Domain:
    - All integer values in the range [0,9]

```
  S E N D
+ M O R E
---------
M O N E Y
```

Variables and domains in CP

- Integer domains
  - Defined by a lower and upper bound
- Logical domains
  - Either *true* or *false*
- Enumeration
  - Represent a set of n possible choices

Variables and domains in CP

- Real domains
  - Continuous range within two bounds – a variable can take any real value within that range
- Finite (or *Set*) domains
  - A variable can take any value from the ones present in the set

Constraints

- Integer and Real variables have arithmetic constraints
  - e.g. $x + y \leq z$
- Logical variables have logical constraints
  - e.g. $(x \vee y) \wedge \sim z$
- Set constraints
  - Enforce element and subset relations between variables

How are CSPs solved?

- Combination of three techniques
  - Domain reduction
  - Constraint propagation
  - Backtracking search

Domain reduction

- Domain reduction is the direct application of a constraint to a variable's domain.
  - e.g., if the domain of x is [0, 10] and a constraint states that $x > 3$, then the domain of x becomes [4, 10]

Constraint propagation

- Constraint propagation is the propagation of changes in a variable's domain to the domain of other variables related by constraints.
  - e.g., x and y have domain [0, 10]
  - $x \le y - 3$
  - y's domain becomes [3, 10]
  - x's domain becomes [0, 7]

Backtracking search

- General algorithm that finds solutions for a given computational problem
  - Incrementally build a candidate solution
    - A set of values for the problem's variables
  - Abandons each partial candidate as soon as it determines that it can't be completed to a valid solution
  - Backtracks to the previous valid partial solution

Constraint Optimization Problems

- Often, just finding a valid solution is not enough. We want to find an **optimal solution**.
  - **Constraint Optimization Problem** – Similar to the Constraint Satisfaction Problem
  - But with an additional **Objective Function**: A function of the problems' variables that specifies a preference between solutions
  - Our goal is to find the solution that maximizes or minimizes the **O.F.**'s value

Why using CP techniques to solve scheduling problems?

A scheduling problem is a constraint satisfaction problem:

- Activities (tasks) = **decision variables**
- Task allocation to resources is limited by **constraints**
- Tasks may only be assigned to certain resources and have certain times for execution – **variable domains**

Problem search space

The search space of the problem is the the space of possible assignment of tasks to resources and the timing of these tasks

How can scheduling problems be modeled as CSPs?

An activity (task) occupies space (resources) and time,

so what should be represented by:

- Variables and domains
- Constraints

Variables and domains

- A **starting time** and a **duration** are the minimum necessary variables to define a task
- An **ending time** should be used if the need for further constraints arises

Scheduling-specific constraints

Different types of scheduling problems:

- Disjuntive scheduling
  – One task per resource at any given time
- Cumulative scheduling
  – A resource can execute several tasks in parallel
- Preemptive and Non-preemptive scheduling
  – Tasks may or may not be interrupted, respectively

Scheduling-specific constraints

- Sequencing of activities
  - Definition of precedence between activies
- Disjuntive constraints
  - Definition of activities that must not overlap

Representing activities (non-preemptive)

For any activity:

- 2 variables: start and end time of an activity
- Lower and upper bounds for those variables
  - EST, LST, EET, LET
- Constraints regarding the duration of the activity may be defined using the difference between the start and the end of the activity

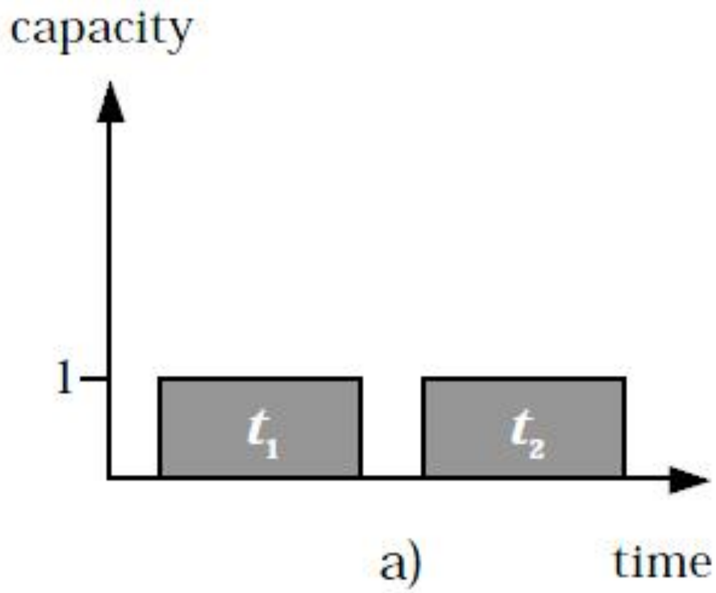
Representing activities (preemptive)

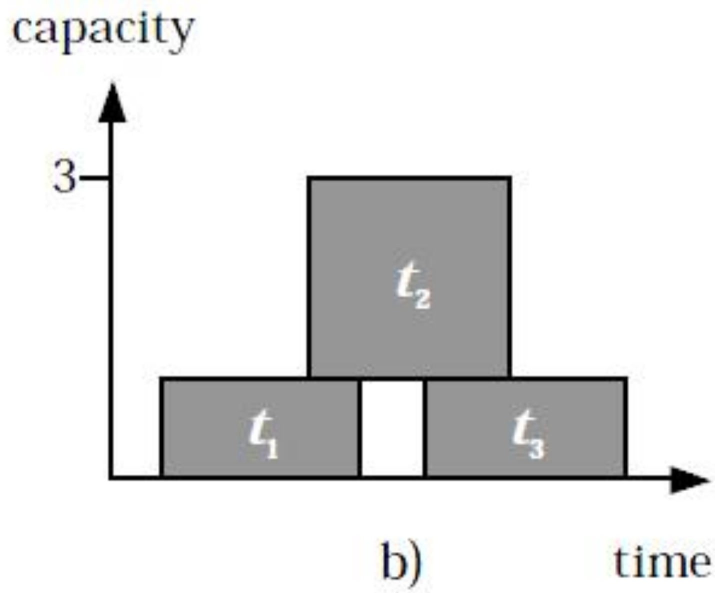
- An activity may be represented using a set
  - Contains the intervals where the activity was executed
  - The duration of the task is given by the elements of the set

Resource types

- Unary resources
  - Can only execute one activity at a given time
- Cumulative resources
  - Can execute several activities in parallel if they have the capacity for it
- Renewable / consumable resources
  - May be depleted in the course of activities

Resource types





a) Activities performed by unary resources

b) Activities performed by cumulative resources

Final Remarks

- Constraint-based scheduling provides a high-level framework that separates implementation details from system description

- Allows for a complete definition of the properties of a solution for a scheduling problem

?

Bibliography

- Constraint-based Scheduling
  - Markus P. J. Fromherz
- Constraint-Based Scheduling: A tutorial
  - Claude Le Pape