2G1515: Constraint Programming

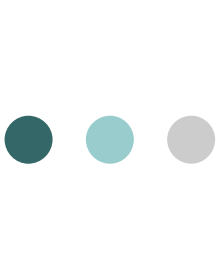
Introduction and Overview

Lecture 01, 2006-01-24

Christian Schulte
schulte@imit.kth.se

ElecSchool of Information and Communication Technology
KTH – Royal Institute of Technology
Stockholm, Sweden

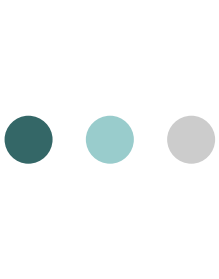
KTH Information and Communication Technology



Plan of Lecture

- Introduction
  - what is constraint programming?
- Overview
  - course content
  - course goal
- Organizational

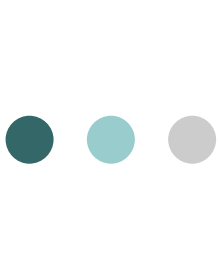
Constraint Programming



Constraint Programming

- solving combinatorial problems

start with a first toy problem



Send More Money (SMM)

- find distinct digits for letters, such that

```
  SEND
+ MORE
-------
= MONEY
```

Constraint Model for SMM

- variables:

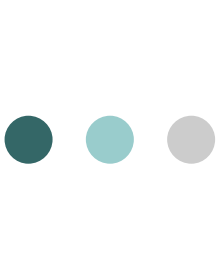
  $S,E,N,D,M,O,R,Y \in \{0,\ldots,9\}$

- constraints:

  $$\texttt{distinct}(S,E,N,D,M,O,R,Y)$$

  $$\begin{array}{rr} & 1000\times S+100\times E+10\times N+D \\ + & 1000\times M+100\times O+10\times R+E \\ = & 10000\times M+1000\times O+100\times N+10\times E+Y \end{array}$$

  $$S\neq 0 \qquad M\neq 0$$



Solving SMM

- find values for variables

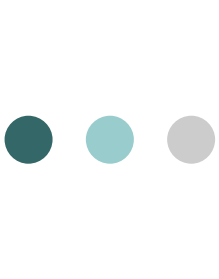
  such that

  all constraints satisfied

Finding a Solution

- enumerate assignments: poor!
- constraint programming
  - compute with possible values
  - prune inconsistent values
    constraint propagation
  - search
    - branch: define search tree
    - explore: explore search tree for solution

Principles and Applications



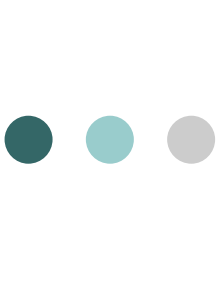
Principles and Applications

- Constraint propagation
- Search
- Summary
- Application example
  - resource scheduling

Constraint Propagation

Important Concepts

- Constraint store
- Basic constraint
- Propagator
- Non-basic constraint
- Constraint propagation



Constraint Store

$x \in \{3,4,5\}$ $y \in \{3,4,5\}$

- Stores basic constraints

  map variables to possible values

Constraint Store

finite domain constraints

$x \in \{3,4,5\}$ $y \in \{3,4,5\}$

- Stores basic constraints

  map variables to possible values

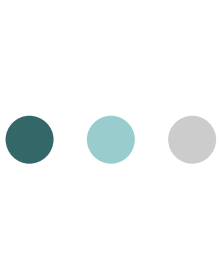
Constraint Store

$x \in \{3,4,5\}$ $y \in \{3,4,5\}$

- Stores basic constraints

  map variables to possible values

- Domains: finite sets, real intervals, trees, …



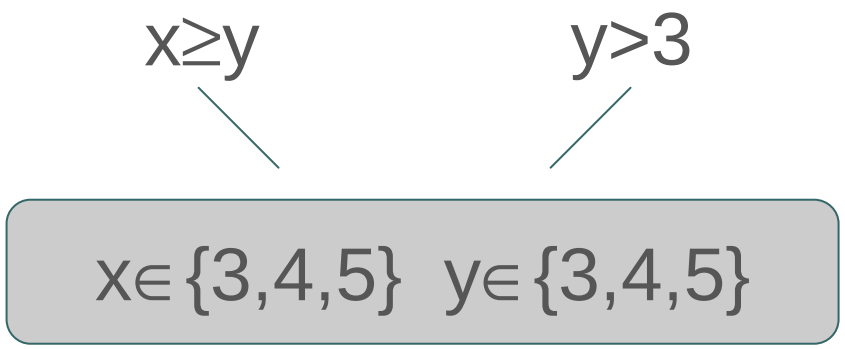
Propagators

- Implement non-basic constraints

$$\texttt{distinct}(x_1,\ldots,x_n)$$

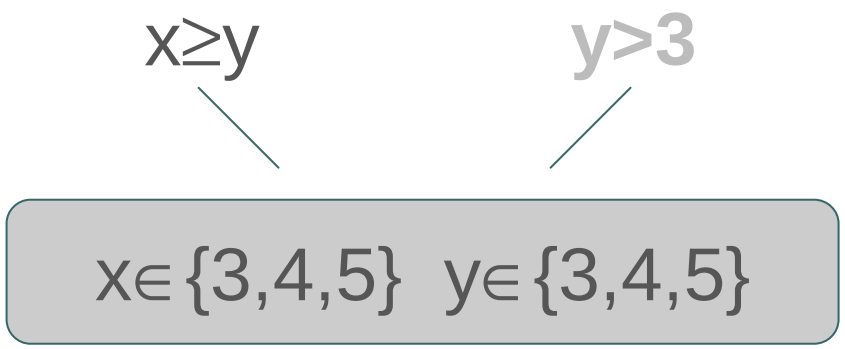
$$x + 2 \times y = z$$

Propagators



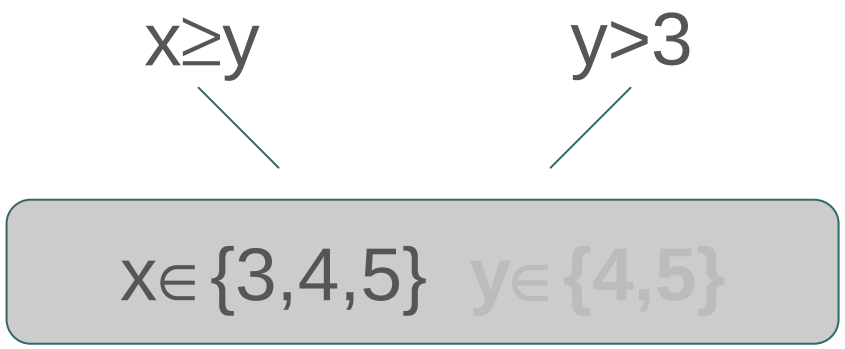
- Amplify store by constraint propagation

Propagators



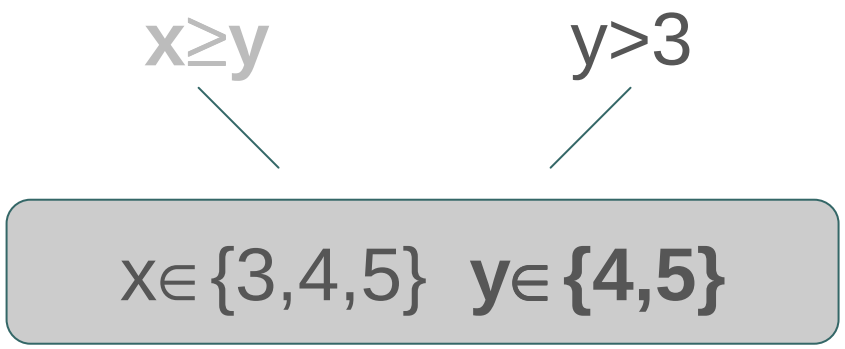
- Amplify store by constraint propagation

Propagators



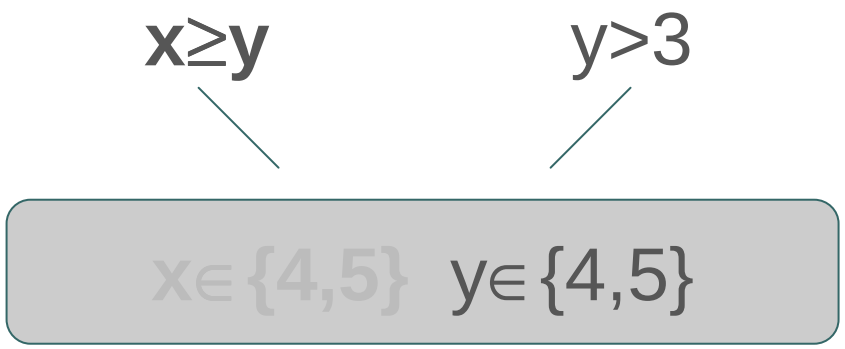
- Amplify store by constraint propagation

Propagators



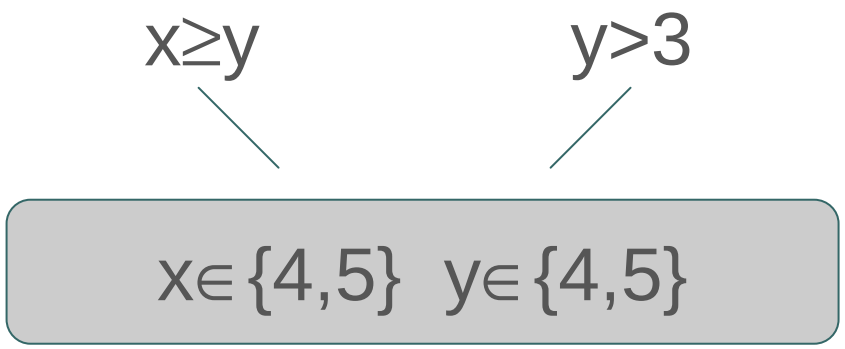
- Amplify store by constraint propagation

Propagators



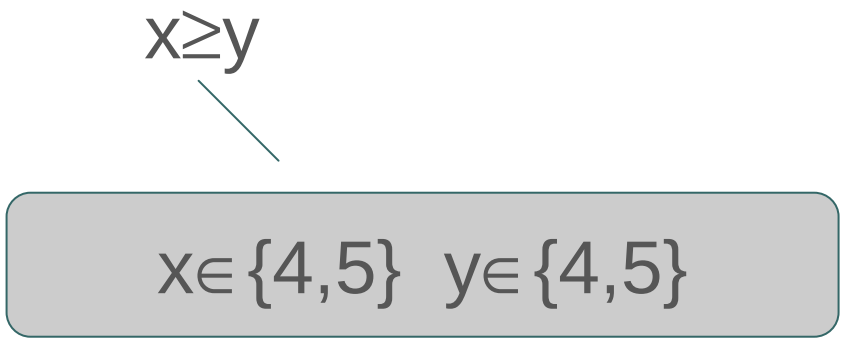
- Amplify store by constraint propagation

Propagators



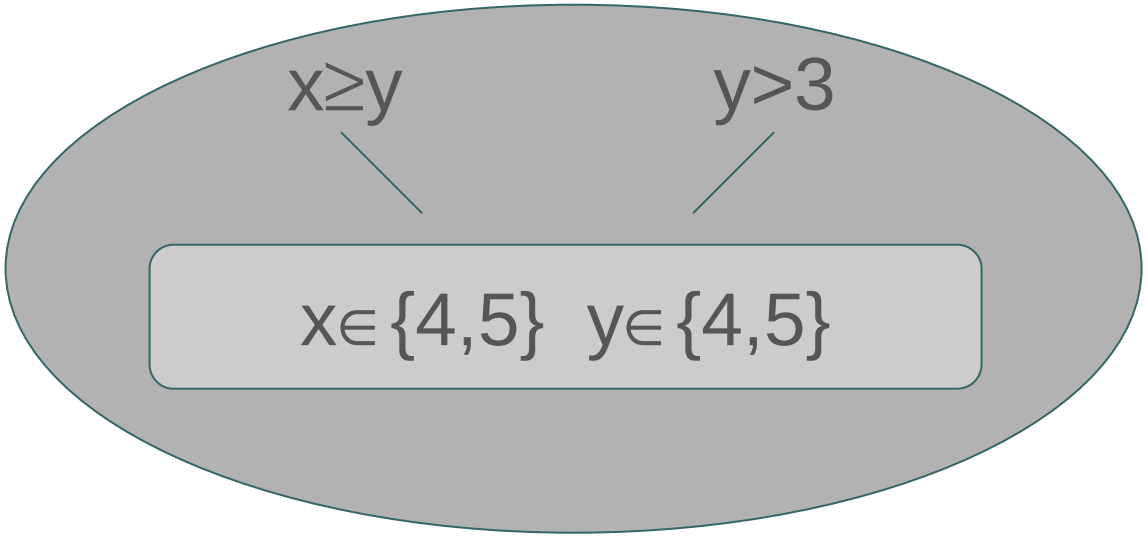
- Amplify store by constraint propagation
- Disappear when done (entailed)
  - no more propagation possible

Propagators



- Amplify store by constraint propagation
- Disappear when done (entailed)
  - no more propagation possible

Computation Space



- Store with connected propagators

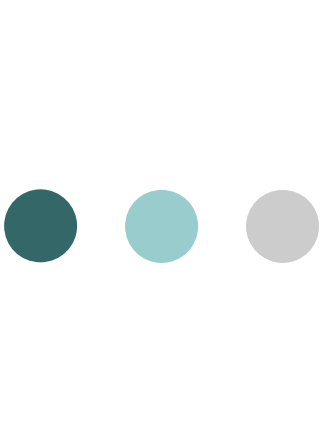
Propagation for SMM

- Results in store

  $S=9 \quad E \in \{4,\ldots,7\} \quad N \in \{5,\ldots,8\} \quad D \in \{2,\ldots,8\}$

  $M=1 \quad O=0 \quad\quad\quad\quad R \in \{2,\ldots,8\} \quad Y \in \{2,\ldots,8\}$

- Propagation **alone** not sufficient!
  - create simpler sub-problems
  - branching

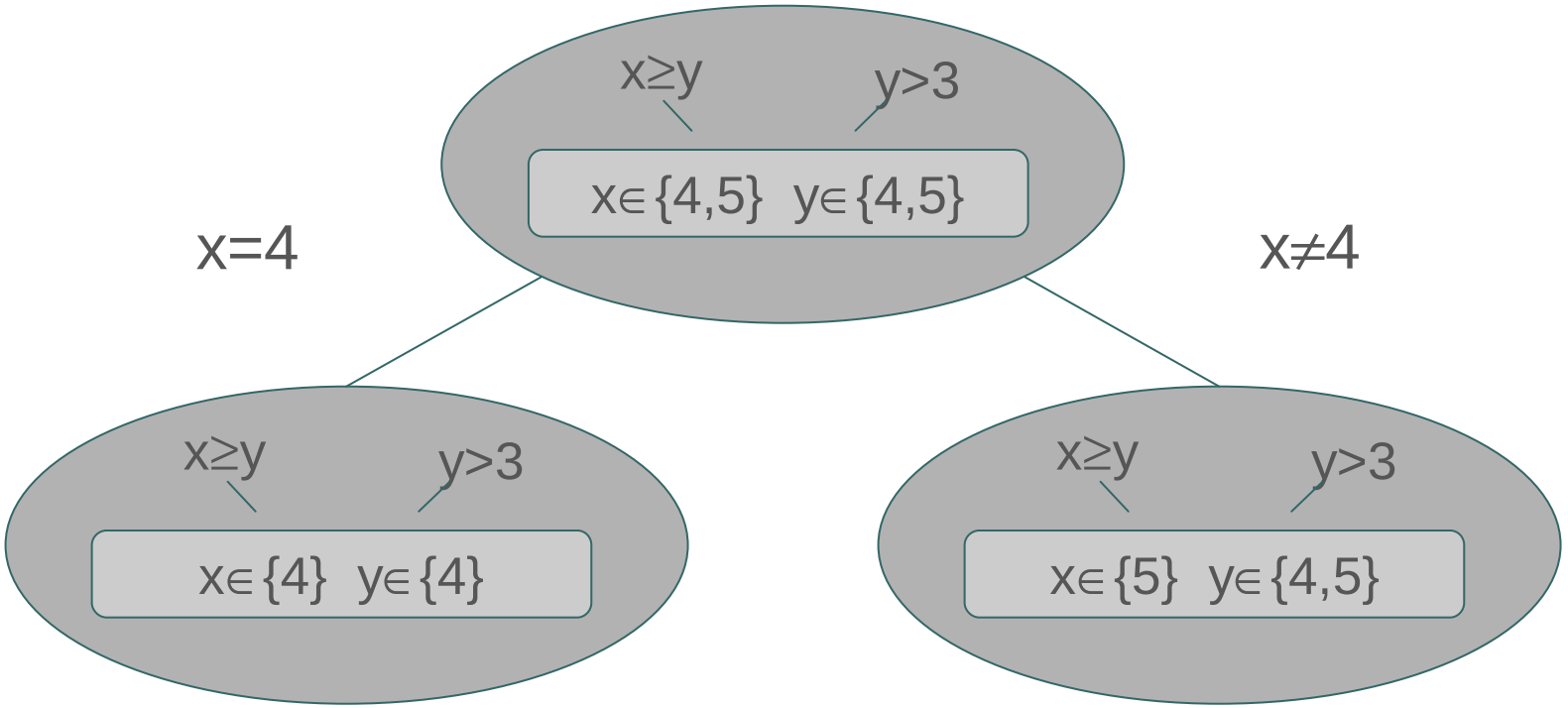


Search

Important Concepts

- Branching
- Exploration
- Branching heuristics
- Best solution search

Branching



- Yields spaces with additional constraints
- Enables further constraint propagation

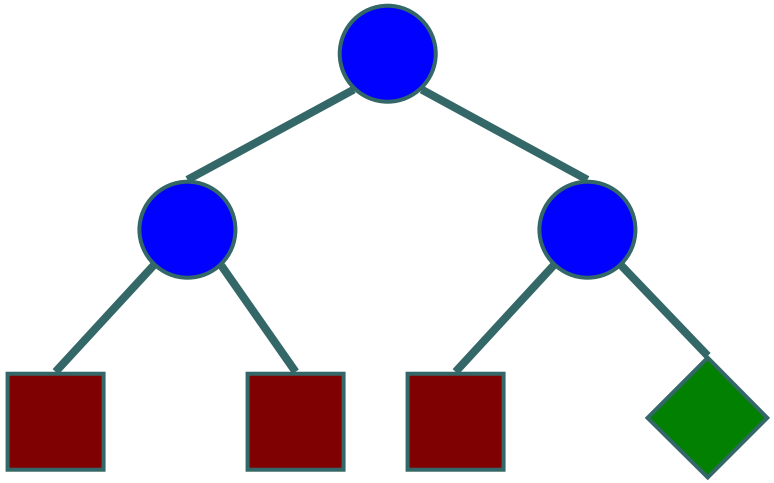
Branching Strategy

- Pick variable x with at least two values
- Pick value n from domain of x
- Branch with

  $x=n$ and $x\neq n$

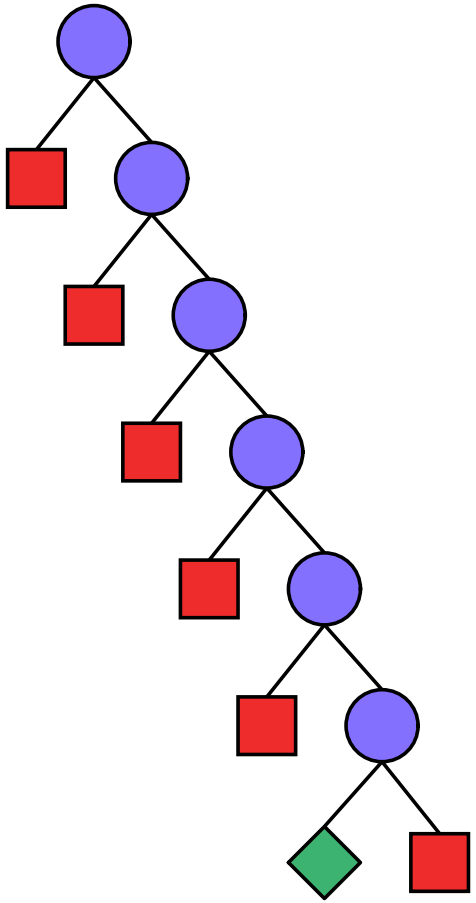
- Part of model

Search



- Iterate propagation and branching
- Orthogonal: branching ⇆ exploration
- Nodes:
  - **Unsolved**
  - **Failed**
  - **Succeeded**

SMM: Solution



```
   SEND
+  MORE
-------
= MONEY
```

```
   9567
+  1085
-------
= 10652
```

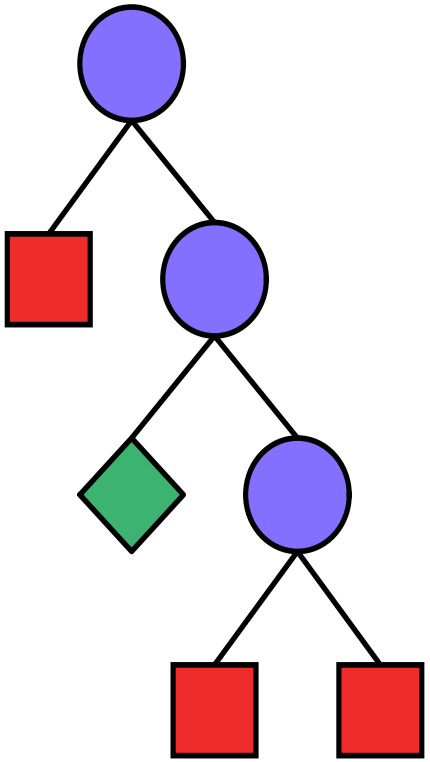
Heuristics for Branching

- Which variable
  - least possible values (first-fail)
  - application dependent heuristic
- Which value
  - minimum, median, maximum

    $x=m$ or $x\neq m$
  - split with median m

    $x<m$ or $x\geq m$
- Problem specific

SMM: Solution With First-fail



```
   SEND
+  MORE
------
= MONEY
```

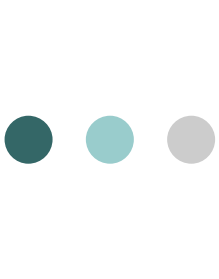
```
   9567
+  1085
------
= 10652
```

Send Most Money (SMM++)

- Find distinct digits for letters, such that

```
  SEND
+ MOST
------
= MONEY
```

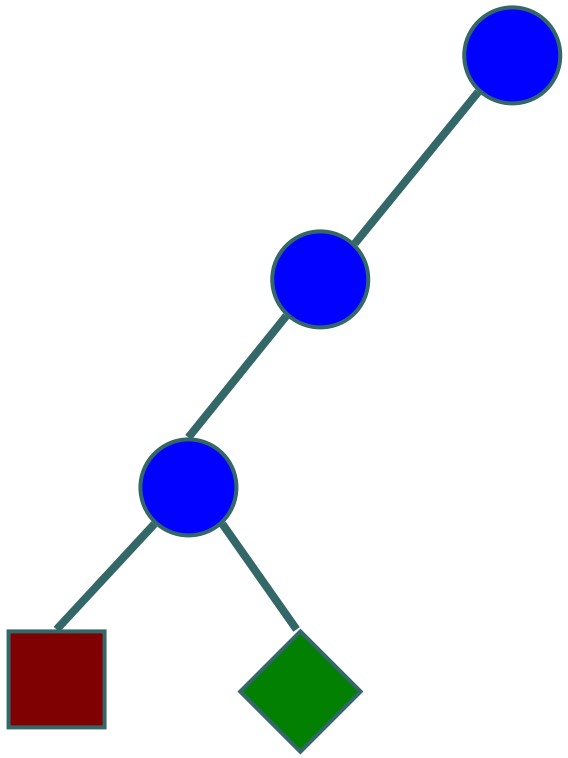
and MONEY maximal



Best Solution Search

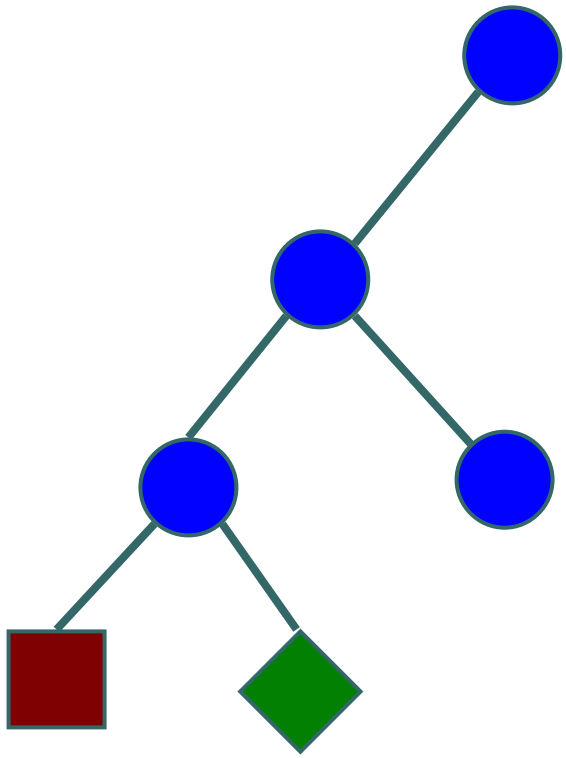
- Naïve approach:
  - compute all solutions
  - choose best
- Branch-and-bound approach:
  - compute first solution
  - add “betterness” constraint to open nodes
  - next solution will be “better”
  - prunes search space

Branch-and-bound Search



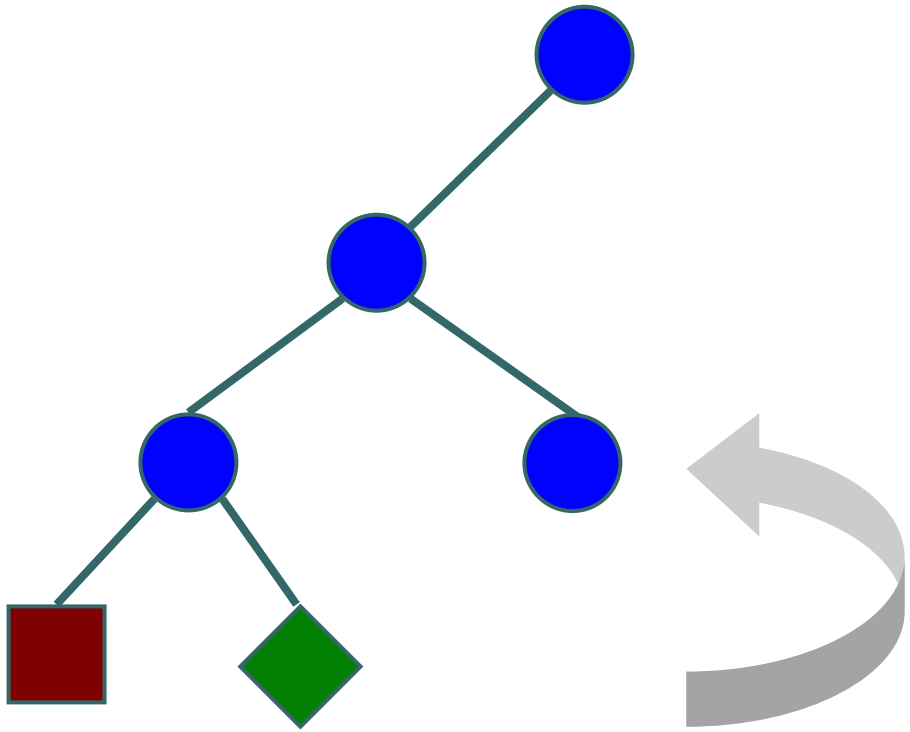
- Find first solution

Branch-and-bound Search



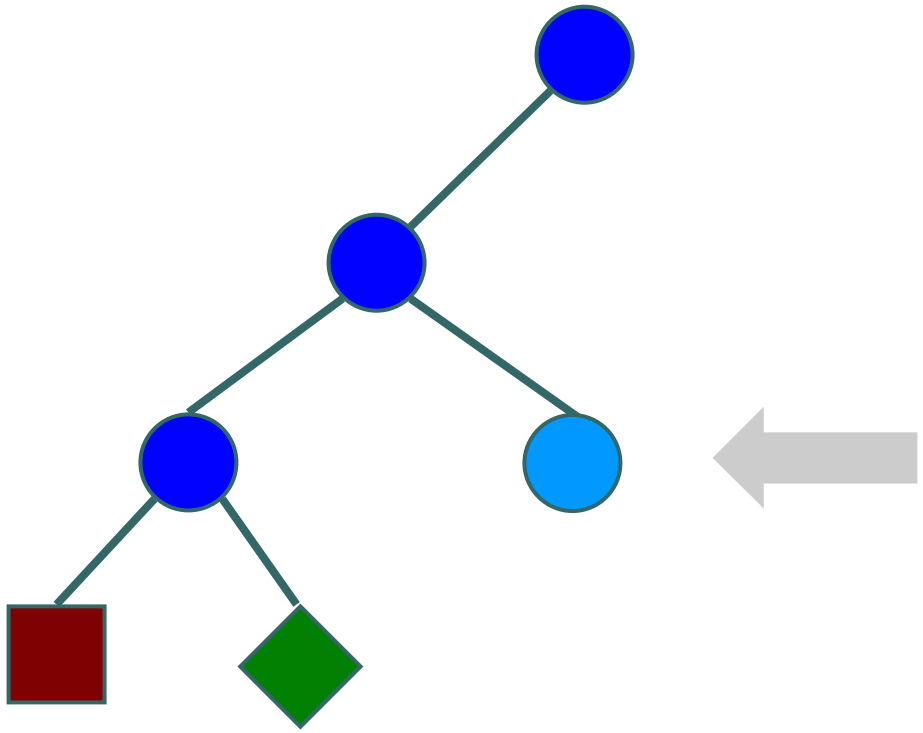
- Explore with additional constraint

Branch-and-bound Search



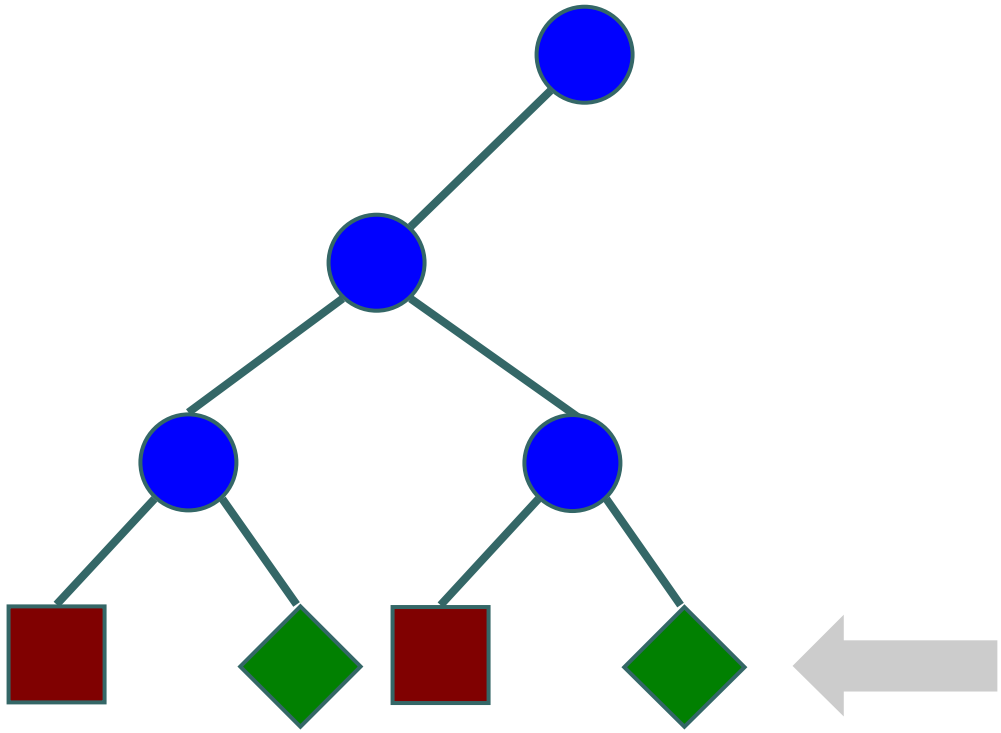
- Explore with additional constraint

Branch-and-bound Search



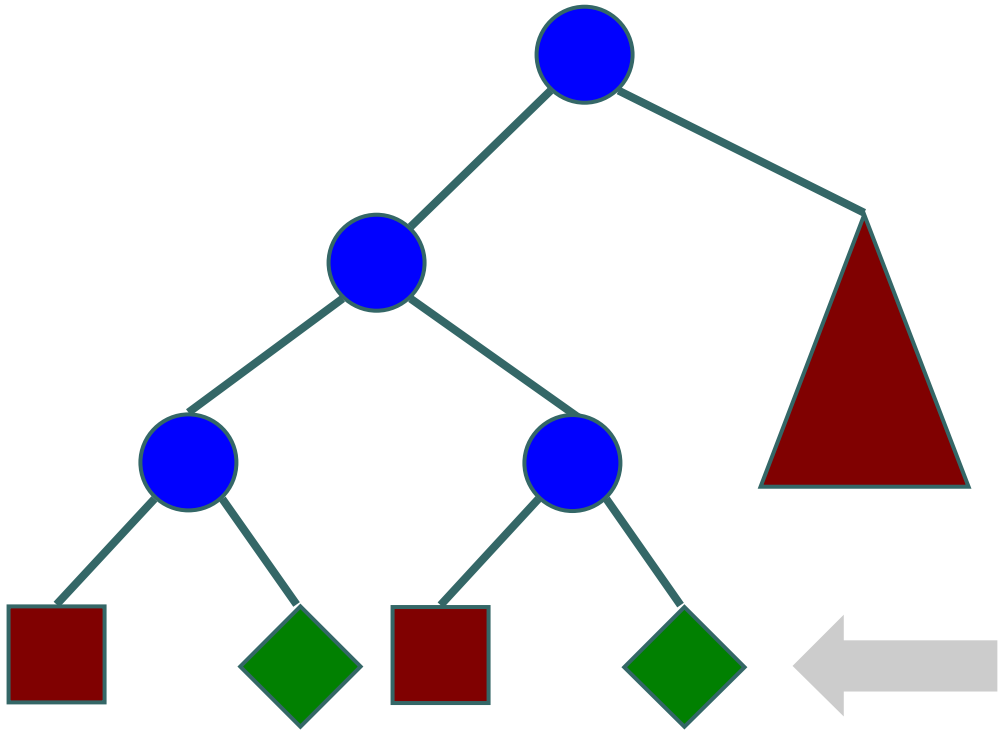
- Guarantees better solutions

Branch-and-bound Search



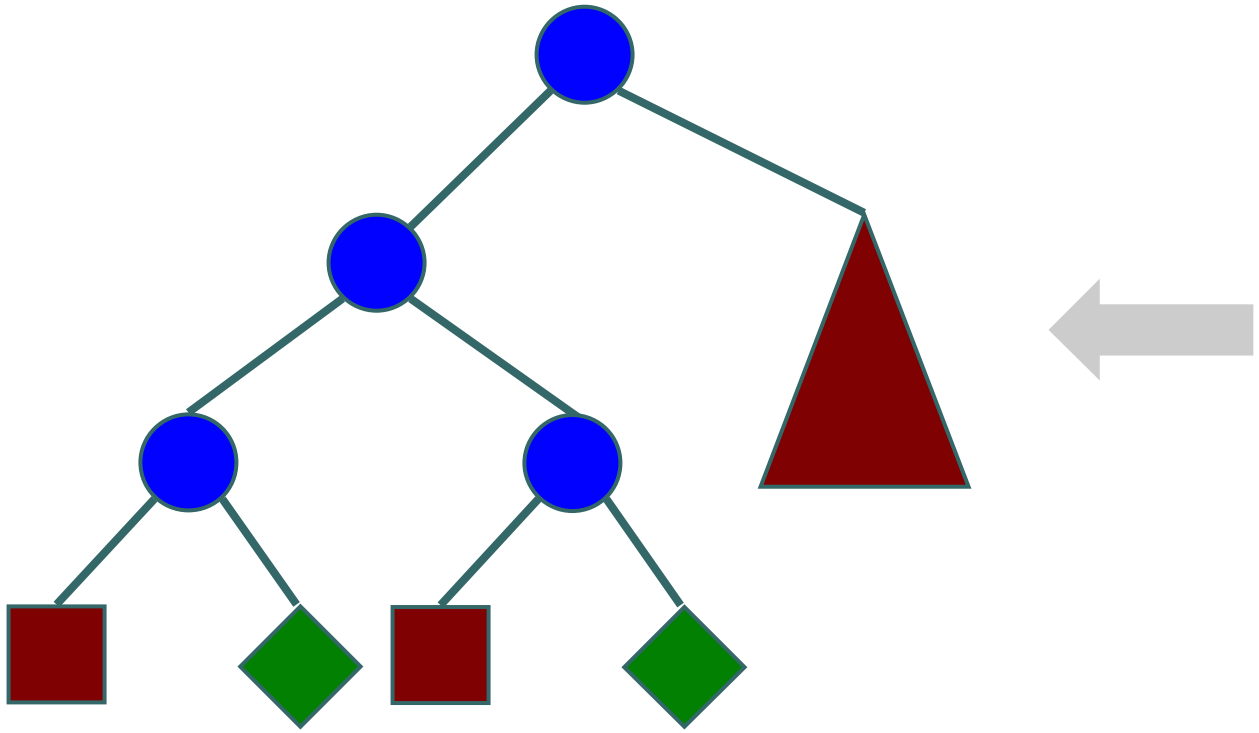
- Guarantees better solutions

Branch-and-bound Search



- Last solution best

Branch-and-bound Search



- Proof of optimality

Modelling SMM++

- Constraints and branching as before
- Order solutions with constraints
  - so-far-best solution

    S,E,N,D,M,O,T,Y
  - current node

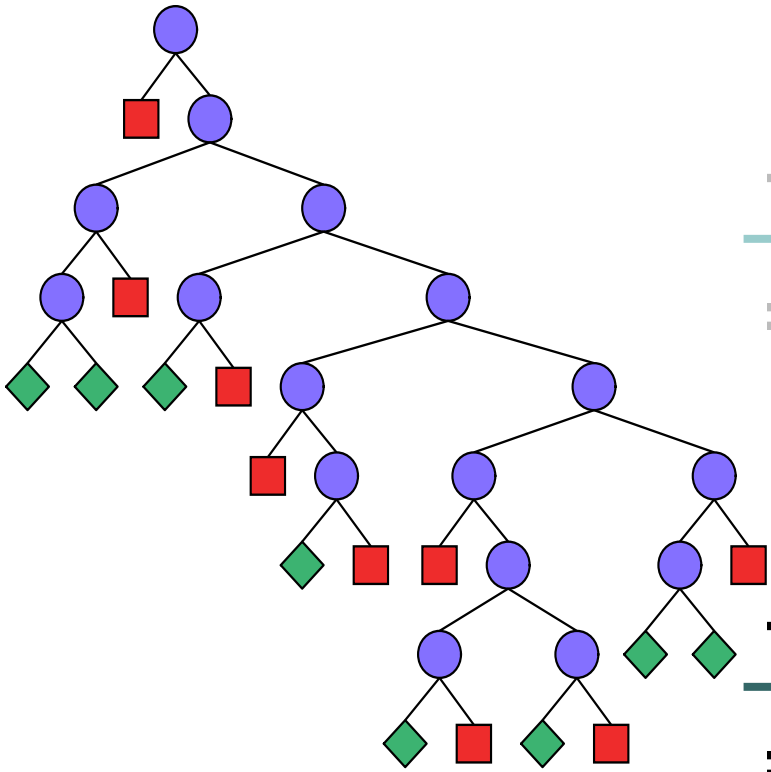
    S,E,N,D,M,O,T,Y
  - constraint added

    $10000\times M+1000\times O+100\times N+10\times E+Y$

    $<$

    $10000\times \mathbf{M}+1000\times \mathbf{O}+100\times \mathbf{N}+10\times \mathbf{E}+\mathbf{Y}$

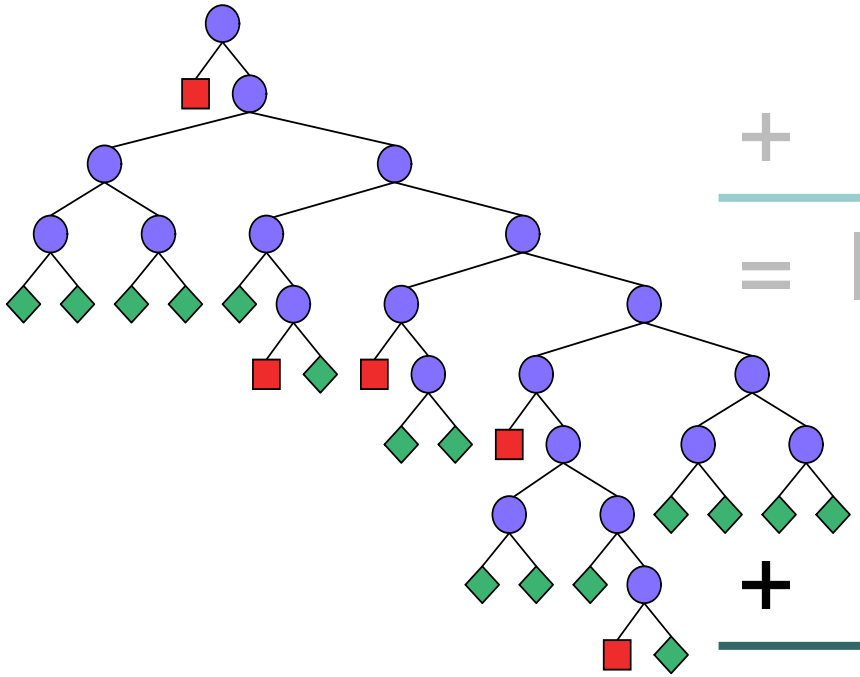
SMM++: Branch-and-bound



```
  SEND
+ MOST
------
= MONEY
```

```
  9782
+ 1094
------
= 10876
```

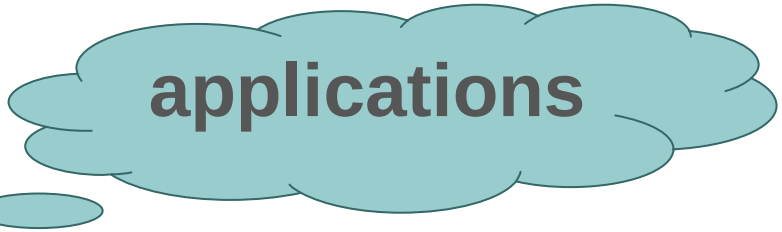
SMM++: All Solution Search



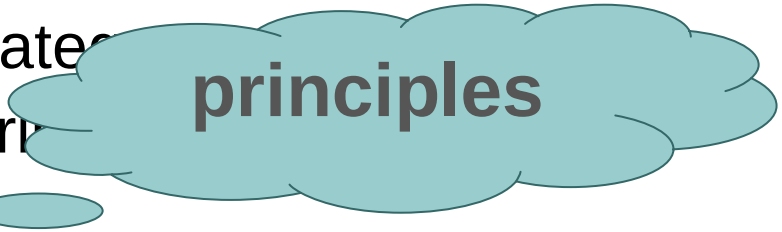
```
  SEND
+ MOST
------
= MONEY

  9782
+ 1094
------
= 10876
```

Summary

Summary

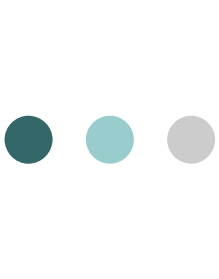


- Modeling
  - variables with domain
  - constraints to state relations
  - branching strate
  - solution order
- Solving
  - constraint propagation
  - constraint branching
  - search tree exploration



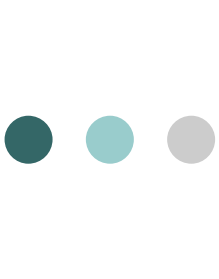
Application Areas

- Timetabling
- Scheduling
- Crew rostering
- Resource allocation
- Workflow planning and optimization
- Gate allocation at airports
- Sports-event scheduling
- Railroad: track allocation, train allocation, schedules
- Automatic composition of music
- Genome sequencing
- Frequency allocation
- …



Application Example

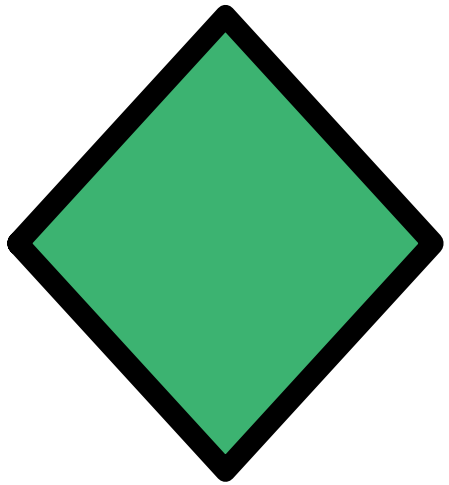
- Scheduling resources
  - machines, personal, …
  - constraint programming showcase



Principles

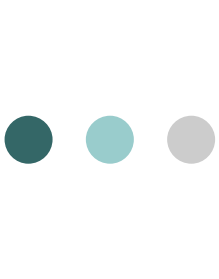
- Models for constraint propagation
  - properties and guarantees
- Strong constraint propagation
  - global constraints with strong algorithmic methods
  - mantra: search kills, search kills, search k…
- Branching strategies
- Exploration strategies

SMM: Strong Propagation



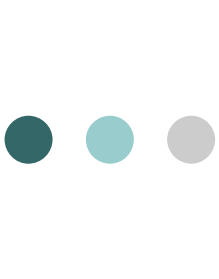
```
   SEND
+  MORE
------
= MONEY
```

```
   9567
+  1085
------
= 10652
```



Used Techniques

- Artificial intelligence
- Operations research
- Algorithms
- Programming languages



Why Does CP Matter?

- Middleware for combining smart algorithmic components
  - scheduling
  - graphs
  - flows
  - …

  plus
  - essential extra constraints

Significance

- Constraint programming identified as a strategic direction in computer science research

  [ACM Computing Surveys, Dec 1996]

- Applications are ubiquitous
  - knowledgeable people are not!

Scheduling Resources

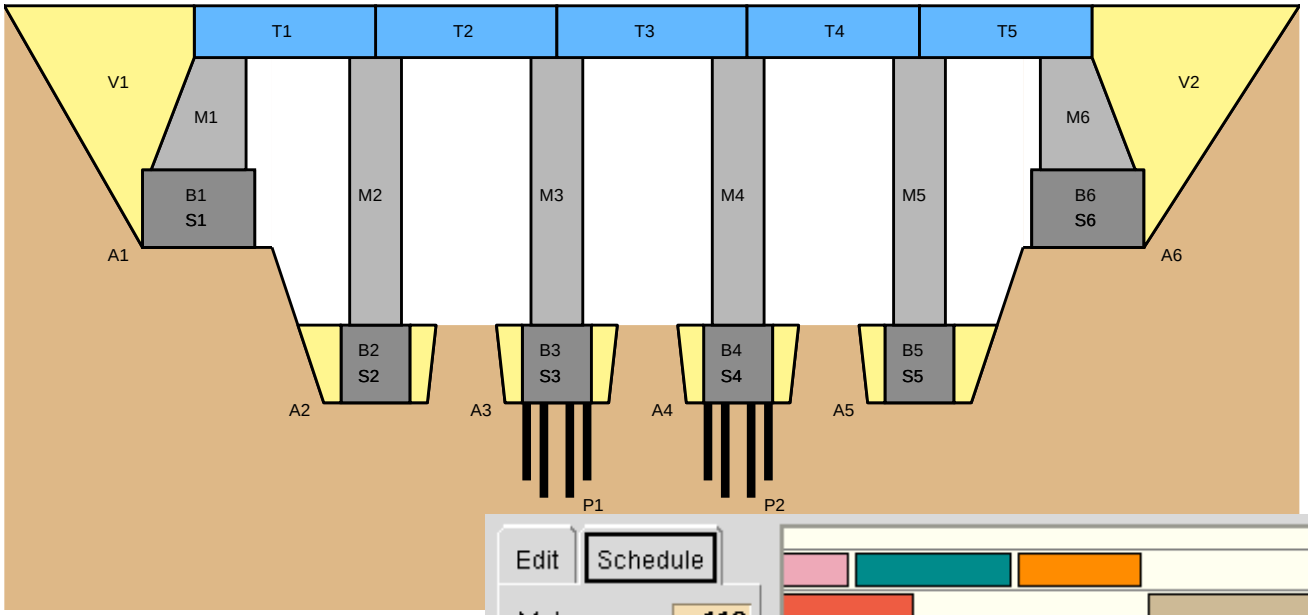
- Modeling
- Propagation
- Strong propagation

Scheduling Resources: Given

- Tasks
  - duration
  - resource
- Precedence constraints
  - determine order among two tasks
- Resource constraints
  - at most one task per resource

    [disjunctive, non-preemptive scheduling]

Scheduling: Bridge Example



Scheduling: Solution

- Start time for each task
- All constraints satisfied
- Earliest completion time
  - minimal make-span

Scheduling: Model

- Variable for start-time of task *a*

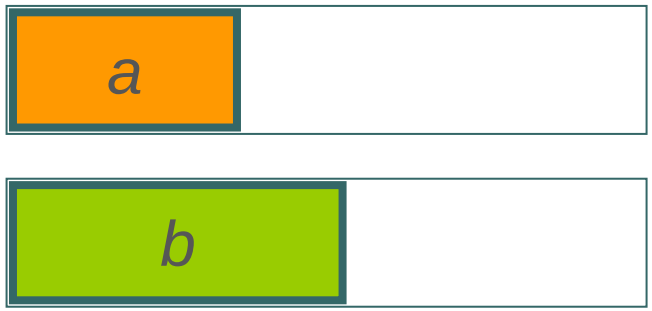
  start(*a*)

- Precedence constraint: *a* before *b*

  start(*a*) + dur(*a*) ≤ start(*b*)

Propagating Precedence

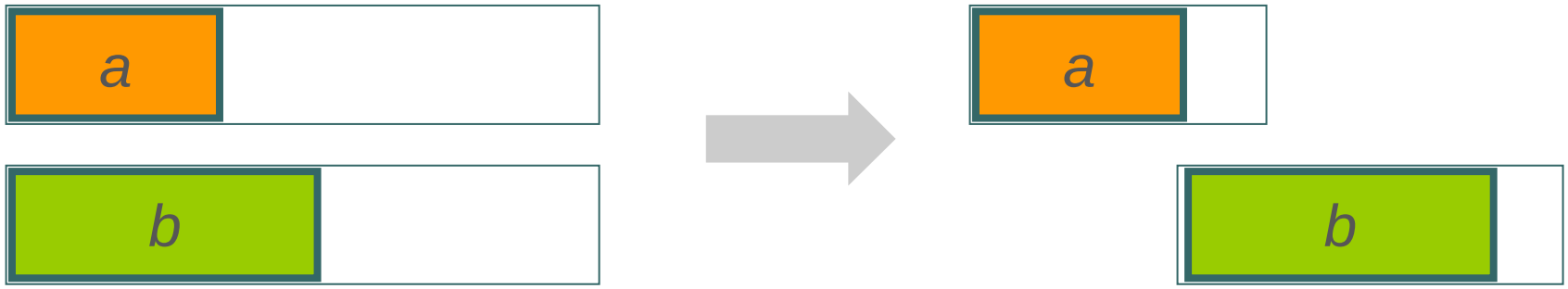
a before *b*



start(*a*)∈{0,…,7}
start(*b*)∈{0,…,5}

Propagating Precedence

a before *b*



start(*a*)∈{0,…,7}
start(*b*)∈{0,…,5}

start(*a*)∈{0,…,2}
start(*b*)∈{3,…,5}

Scheduling: Model

- Variable for start-time of task *a*

  start(*a*)

- Precedence constraint: *a* before *b*

  start(*a*) + dur(*a*) ≤ start(*b*)

- Resource constraint:

  a before *b*

  or

  b before *a*

Scheduling: Model

- Variable for start-time of task *a*

  start(*a*)

- Precedence constraint: *a* before *b*

  start(*a*) + dur(*a*) ≤ start(*b*)

- Resource constraint:

  start(*a*) + dur(*a*) ≤ start(*b*)

  or

  b before *a*

Scheduling: Model

- Variable for start-time of task *a*

  start(*a*)

- Precedence constraint: *a* before *b*

  start(*a*) + dur(*a*) ≤ start(*b*)

- Resource constraint:

  start(*a*) + dur(*a*) ≤ start(*b*)

  or

  start(*b*) + dur(*b*) ≤ start(*a*)

Reified Constraints

- Use control variable $b \in \{0,1\}$

$$c \quad \leftrightarrow \quad b=1$$

- Propagate
  - c holds $\Rightarrow$ propagate $b=1$
  - $\neg c$ holds $\Rightarrow$ propagate $b=0$
  - $b=1$ holds $\Rightarrow$ propagate c
  - $b=0$ holds $\Rightarrow$ propagate $\neg c$

Reification for Disjunction

- Reify each precedence

  $[\text{start}(a) + \text{dur}(a) \leq \text{start}(b)] \leftrightarrow b_0 = 1$

  and

  $[\text{start}(b) + \text{dur}(b) \leq \text{start}(a)] \leftrightarrow b_1 = 1$

- Model disjunction

  $b_0 + b_1 \geq 1$

Model Is Too Naïve

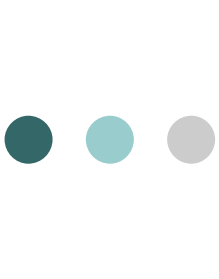
- Local view
  - individual task pairs
  - $O(n^2)$ propagators for n tasks
- Global view
  - all tasks on resource
  - single propagator
  - smarter algorithms possible

Edge Finding

- Find ordering among tasks (“edges”)
- For each subset of tasks $\{a\} \cup B$
  - assume: *a* before *B*

    deduce information for *a* and *B*
  - assume: *B* before *a*

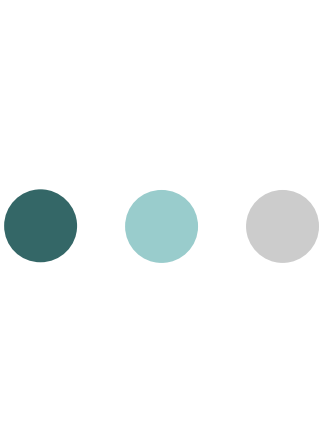
    deduce information for *a* and *B*
  - join computed information
  - can be done in $O(n^2)$



Summary

- Modeling
  - easy but not always efficient
  - constraint combinators (reification)
  - global constraints
  - smart heuristics
- More on constraint-based scheduling

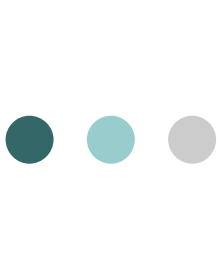
  Baptiste, Le Pape, Nuijten. Constraint-based Scheduling, Kluwer, 2001.



Course Overview

Content Overview

- As to be expected, no surprises:
  - **applications**
  - **principles**
  - pragmatics
  - limitations

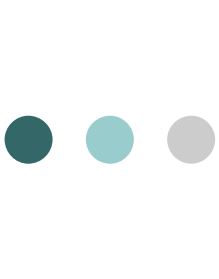


Principles

- Models for and properties of
  - constraint propagation
  - search
  - combination mechanisms
- Study and overview of
  - propagation algorithms
  - search heuristics
  - exploration algorithms
  - constraint programming systems

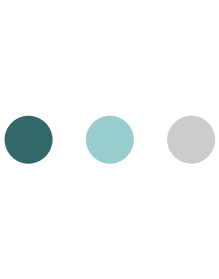
Applications

- Modeling techniques
  - symmetries
  - heuristics
  - algorithm selection
- Application areas
  - scheduling
  - assignment



Pragmatics

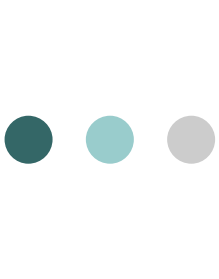
- Using constraint programming in practice
- Apply knowledge on
  - principles
  - applications
- Understand limitations



Goal

- Basic understanding of constraint programming
  - applications
  - principles
- Skills to apply in practice

Organizational



Material

- Lecture notes (slides)
  - available before the lectures…
- Additional material
  - book excerpts
  - scientific articles
  - notes written by me

How to Pass?

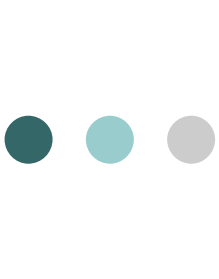
- Pass exam
  - has 240 (4 hour exam) exam points
  - 120 total points needed
  - grading scale linear (see www)
- Total pts = exam pts + bonus pts
- Bonus points
  - from assignments
  - at most 40 points

Assignments

- Four assignments
  - each 10 bonus points if submitted in time
  - one week for solving
- Points only valid in this academic year!

Assignment Tasks

- Exploration tasks
  - small tryouts
  - need to be done in order to do…
- Submission tasks
  - submit in time, get bonus points
  - do them, do them
- Both practical and principles

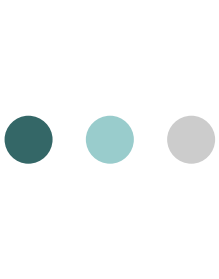


Software: Gecode/J

- Java frontend to Gecode C++ library
- You will be guinea pigs…
- Course webpage
  - packages
  - installation information

Contacting…

- Christian Schulte
  - schulte@imit.kth.se
  - other options, see my homepage
- Mikael Lagerkvist (assignments)
  - zayenz@kth.se
- You and me
  - mailing list: subscribe as on webpage
  - you ask… everybody can answer…



Summary

- Constraint programming…
  - …is exciting!
  - …is fun!
- Understanding of principles and applications necessary
- Read the webpage
  - **`www.imit.kth.se/courses/2G1515/`**