Scheduling and Assignment

Miodrag Potkonjak
Scheduling

- Complexity
- Problem Solving Techniques
- Directions
Scheduling using Simulated Annealing

Reference:
Devadas, S.; Newton, A.R.

Algorithms for hardware allocation in data path synthesis.

Simulated Annealing

- Local Search

Cost function vs. Solution space
State \{r:\}  (configuration -- a set of atomic position)

weight \ e^{-E(\{r:\})/K_B T}  -- Boltzmann distribution

E(\{r:\}): energy of configuration

K_B: Boltzmann constant

T: temperature

Low temperature limit ??
Analogy

Physical System | Optimization Problem
--- | ---
State (configuration) | Solution
Energy | Cost function
Ground State | Optimal solution
Rapid Quenching | Iteration improvement
Careful Annealing | Simulated annealing
Generic Simulated Annealing Algorithm

1. Get an initial solution $S$
2. Get an initial temperature $T > 0$
3. While not yet “frozen” do the following:
   3.1 For $1 \leq i \leq L$, do the following:
      3.1.1 Pick a random neighbor $S'$ of $S$
      3.1.2 Let $\Delta = \text{cost}(S') - \text{cost}(S)$
      3.1.3 If $\Delta \leq 0$ (downhill move) set $S = S'$
      3.1.4 If $\Delta > 0$ (uphill move)
         set $S = S'$ with probability $e^{-\Delta/T}$
   3.2 Set $T = rT$ (reduce temperature)
4. Return $S$
Basic Ingredients for S.A.

- Solution space
- neighborhood Structure
- Cost function
- Annealing Schedule
Integer Linear Programming

- **Given**: integer-valued matrix $A_{mxn}$, vectors $B = (b_1, b_2, ..., b_m)$, $C = (c_1, c_2, ..., c_n)$

- **Minimize**: $C^TX$

- **Subject to**:
  
  $AX \leq B$

  $X = (x_1, x_2, ..., x_n)$ is an integer-valued vector
**Integer Linear Programming**

- Problem: For a set of (dependent) computations \( \{t_1, t_2, \ldots, t_n\} \), find the minimum number of units needed to complete the execution by \( k \) control steps.

- Integer linear programming:
  Let \( y_0 \) be an integer variable.

  For each control step \( i \) (1 \( \leq \) \( i \) \( \leq \) \( k \)):

  define variable \( x_{ij} \) as

  \[ x_{ij} = 1, \text{ if computation } t_j \text{ is executed in the } i \text{th control step.} \]

  \[ x_{ij} = 0, \text{ otherwise.} \]

  define variable \( y_i = x_{i1} + x_{i2} + \ldots + x_{in} \).
Integer Linear Programming

- Integer linear programming:
  For each computation dependency: \( t_i \) has to be done before \( t_j \), introduce a constraint:

\[
k \cdot x_{1i} + (k-1) \cdot x_{2i} + \ldots + x_{ki} < k \cdot x_{1j} + (k-1) \cdot x_{2j} + \ldots + x_{kj} + 1 \quad (*)
\]

Minimize: \( y_0 \)

Subject to:

- \( x_{1i} + x_{2i} + \ldots + x_{ki} = 1 \) for all \( 1 \leq i \leq n \)
- \( y_j \leq y_0 \) for all \( 1 \leq i \leq k \)
- all computation dependency of type (*)
An Example

6 computations
3 control steps
An Example

- Introduce variables:
  - $x_{ij}$ for $1 \leq i \leq 3$, $1 \leq j \leq 6$
  - $y_i = x_{i1} + x_{i2} + x_{i3} + x_{i4} + x_{i5} + x_{i6}$ for $1 \leq i \leq 3$
  - $y_0$

- Dependency constraints: e.g. execute $c_1$ before $c_4$
  - $3x_{11} + 2x_{21} + x_{31} < 3x_{14} + 2x_{24} + x_{34} + 1$

- Execution constraints:
  - $x_{1i} + x_{2i} + x_{3i} = 1$ for $1 \leq i \leq 6$
An Example

- Minimize: \( y_0 \)
- Subject to: \( y_i \leq y_0 \) for all \( 1 \leq i \leq 3 \)
- One solution: \( y_0 = 2 \)
  \( x_{11} = 1, \ x_{12} = 1, \)
  \( x_{23} = 1, \ x_{24} = 1, \)
  \( x_{35} = 1, \ x_{36} = 1. \)
  All other \( x_{ij} = 0 \)