CSE/ECE 848 Introduction to Evolutionary Computation

Module 1 - Lecture 3 - Part 2

Problem Solving and Search: Search

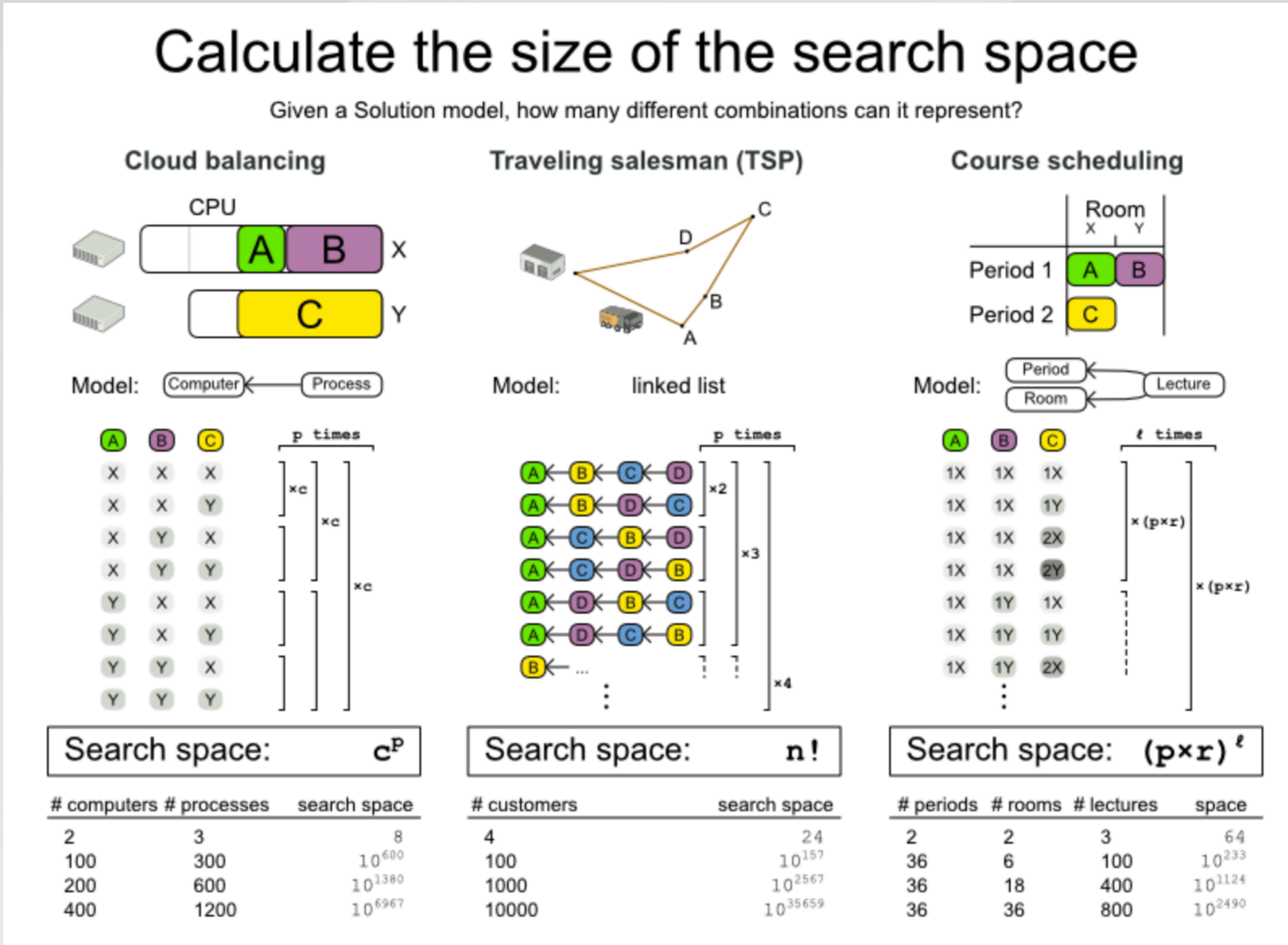
Wolfgang Banzhaf, CSE
John R. Koza Chair in Genetic Programming

Search

- Simulation (single-scenario) is different from optimization/ modelling -> straightforward in the direction of processing
- Multi-scenario simulation as a means to understand a problem

- Optimization & modelling have to search through a huge set of possibilities we call a "space"
- Each solution (or possibility) is a point in the search space
- How large is this space?

Search II



# computers	# processes	search space
2	3	8
100	300	10^{600}
200	600	10^{1380}
400	1200	10^{6967}

# customers	search space
4	24
100	10^{157}
1000	10^{2567}
10000	10^{35659}

# periods	# rooms	# lectures	space
2	2	3	64
36	6	100	10^{233}
36	18	400	10^{1124}
36	36	800	10^{2490}

Complexity

Function	Function Value			
$logN$	1	1.699	2	3
N	10	50	100	1,000
$NlogN$	23.026	765.2	460.52	6,907.75
N^2	100	2,500	10,000	10^6
N^3	1,000	125,000	10^6	10^9
2^N	1,024	1.126×10^{15}	1.27×10^{30}	1.05×10^{301}
10^N	10^2	10^{50}	10^{100}	$10^{1,000}$
$N!$	$3,628.8 \times 10^3$	3.041×10^{64}	10^{158}	4×10^{2567}

From: Chopard/Tomassini, *An Introduction to Metaheuristics for Optimization*, Springer, 2018

Optimization vs. Constraint Satisfaction

- Objective function: a way of assigning a value to a possible solution that reflects its quality on a scale
  - Number of un-checked queens (maximize)
  - Length of a tour visiting given set of cities (minimize)

- Constraint: a binary evaluation telling whether a given requirement holds or not
  - Find a configuration of eight queens on a chessboard such that no two queens check each other
  - Find a tour with minimal length where city X is visited after city Y

Problem vs. Problem Solver

- There is a distinction between the problem (search space) …
- … and the problem solver (mover through search space)

NP Problems

- We only looked at classifying the problem, and did not discuss problem solvers
- This classification scheme needs the properties of the problem solver
- Benefit of this scheme: possible to tell how difficult the problem is

NP - Key Notions

- Problem size: dimensionality of the problem at hand and number of different values for the problem variables

- Running-time: number of operations the algorithm takes to terminate
  - Worst-case as a function of problem size
  - Polynomial or super-polynomial (e.g., exponential)

- Problem reduction: transforming current problem into another via mapping

NP Problems - Classes

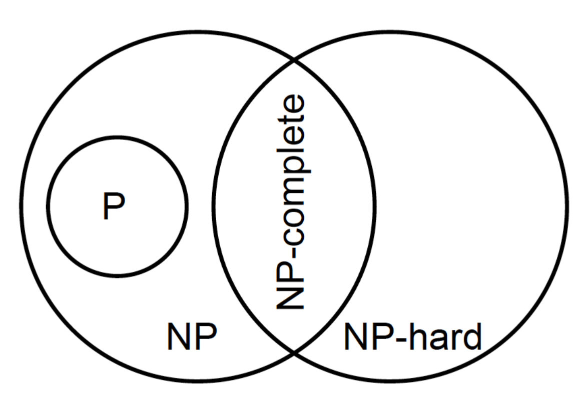
The difficulty of a problem can now be classified:

- Class P: Algorithm can solve the problem in polynomial time (worst-case running-time for problem size n is less than F(n) for some polynomial formula F)
- Class NP: Problem can be solved and any solution can be verified within polynomial time by some other algorithm (P subset of NP)
- Class NP-complete: Problem belongs to class NP and any other problem in NP can be reduced to this problem by an algorithm running in polynomial time
- Class NP-hard: Problem is at least as hard as any other problem in NP-complete but solution cannot necessarily be verified within polynomial time

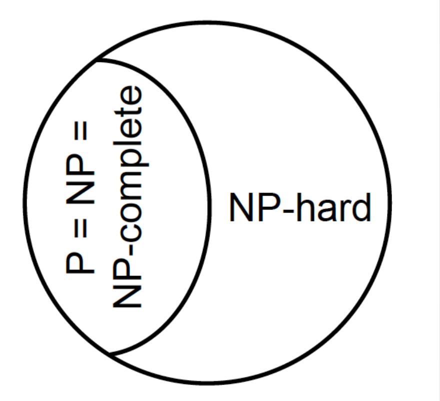
Difference of Classes

- P is different from NP-hard
- Not known whether P is different from NP

P ≠ NP



P = NP



- For now: Use of approximation algorithms and metaheuristics