Search Methods
An Annotated Overview

Edward Tsang

Search Methods – Summary

Systematic Search
- If solutions exist, they will be found
- Specialized methods
  - E.g. Linear Programming
- Branch and Bound (originated from OR)
- A* (started in AI)
- Constraint satisfaction

Stochastic Search
- Sacrifice completeness
- Generate and Test
  - Local Search, e.g.
    - Hill Climbing
    - Simulated Annealing
    - Tabu Search
    - Guided Local Search
  - Artificial Neural Networks
  - Evolutionary computation
    - GA / GP / Ants Colony / …
    - EDA – statistical based

Solution Space & Search Space

17/12/2010

Linear Programming (LP)

- To maximize/minimize an objective function
- Subject to a set of linear inequalities
- Example: to maximize $x + y$
- Subject to:

$2x + y \leq 10$

$x + 2y \leq 12$

$3x - y \leq 9$

17 December 2010

Linear Programming Example

- Maximize $x + y$
- Subject to:

$2x + y \leq 10$

$x + 2y \leq 12$

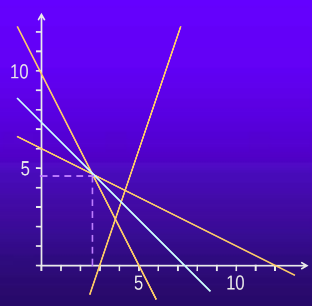
$3x - y \leq 9$

- Solution:

$x = 2.6667$

$y = 4.6667$

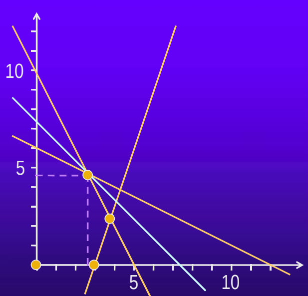
$x + y = 7.3333$



17 December 2010

Simplex Method for LP

- Start with any feasible solution
- Craw along any edge to improve according to objective function
- Repeat until no improvement is available



17 December 2010

Integer Programming

- Maximize $R = 8x_1 + 11x_2 + 6x_3 + 4x_4$
- Subject to $5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14$
  - Where $x_1, x_2, x_3, x_4 \in \{0, 1\}$
- Without integer constraint

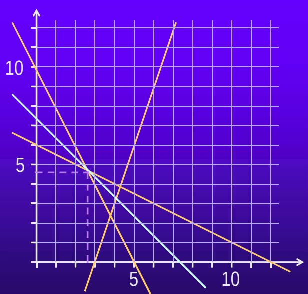
$x_1=1, x_2=1, x_3=0.5, x_4=0$, R = \$22,000

- If we round x_3 down to 0, R = \$19,000
- Better result: $x_1=0, x_2=x_3=x_4=1$, R = \$21,000
- Integer constraint leads to harder problem

17 December 2010

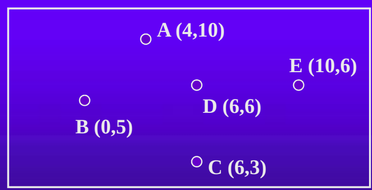
Integer Programming Example

- Maximize x + y
- Subject to:
  - $2x + y \le 10$
  - $x + 2y \le 12$
  - $3x - y \le 9$
- Not a solution:
  - x = 2.6, y = 4.7
- Solutions
  - x = 2, y = 5; x = 3, y = 4



17 December 2010 All Rights Reserved, Edward Tsang

Travelling Salesman Problem (TSP)



- Goal: to find *shortest route* through all cities
- Optimization involved: minimization

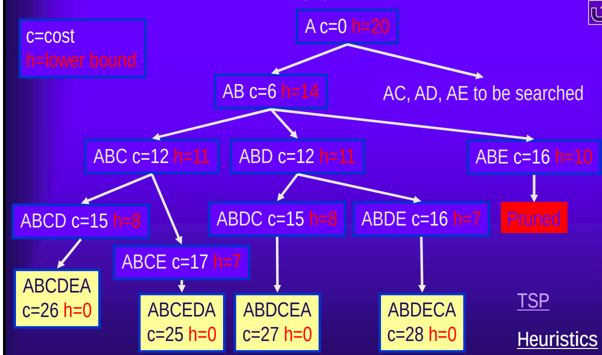
Friday, 17 December 2010 9 Edward Tsang (Copyright)

Distance Table for an example TSP

	A	B	C	D	E
A	--	6	7	4	7
B	6	--	6	6	10
C	7	6	--	3	5
D	4	6	3	--	4
E	7	10	5	4	--
Heuristic:	*4*	*6*	*3*	*3*	*4*

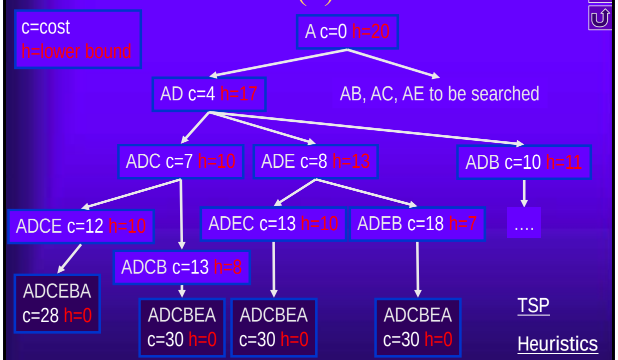
Friday, 17 December 2010 10 Edward Tsang (Copyright)

Branch & Bound (1)



Friday, 17 December 2010 11 Edward Tsang (Copyright)

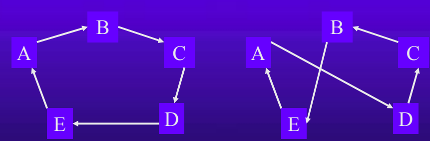
Branch & Bound (2)



Friday, 17 December 2010 12 Edward Tsang (Copyright)

HC Example: 2-opting for TSP

- Candidate tour: a round trip route
- Neighbour: exchange two edges, change directions accordingly



Friday, 17 December 2010 13 Edward Tsang (Copyright)

List reversing ➔ 2-Opting

- List representation:
  - A list could represent cities in sequence
- 2-Opting can be seen as sub-list reversing
  - Easy to implement

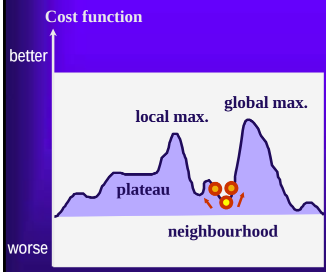
1	3	4	8	6	5	2	7

Breaking points

1	3	4	5	6	8	2	7

Edward Tsang (Copyright) 14 Friday, 17 December 2010

Local Search



- Ingredients:
  - Cost function
  - Neighbourhood function
  - [Optional] Strategy for visiting neighbours
    - e.g. *steepest ascent*
- Problems:
  - local optimum
  - Plateau
  - When to stop?
    - Ok with satisfiability
    - But not optimization

Edward Tsang (Copyright) 15 15 July 2005

GLS: Augmented Cost Function

- Identifying solution ***features***, e.g. Edges used
- associate ***costs*** and ***penalties*** to features
- Given cost function g to minimize
- Augmented Cost Function

λ is parameter to GLS

$$h(s) = g(s) + \lambda \times \Sigma (p_i \times I_i(s))$$

p_i is penalty for feature i, initialized to 0

$I_i(s)$ = 1 if s exhibits feature i; 0 otherwise

15 July 2005 16 Edward Tsang (Copyright)

GLS on TSP

Features:
- n^2 Features
- cost = distance given
- e.g. tour [1,5,3,4,6,2]

- Local search: 2-opting

$$\lambda = a \times g(t^*) / n$$

a = parameter to tune, within (0, 1]

n = # of cities

t* = first local minimum produced by local search; g(t*) = cost of t*

	1	2	3	4	5	6
1					X	
2	X					
3				X		
4						X
5			X			
6		X				

15 July 2005 17 Edward Tsang (Copyright)

Population-based Incremental Learning (PBIL)

Model M:
x = v1 (0.5) 0.6
x = v2 (0.5) 0.4
y = v3 (0.5) 0.4
y = v4 (0.5) 0.6

→ Sample from M solution X, eg <x,v1><y,v4> → Evaluation X

Modify the probabilities

- Statistical approach
  - Related to ant-colonies, GA
- General form is EDA (Estimation of Distribution Algorithms)
  - Build Bayesian Nets (probability dependency)

17 December, 2010 18 Copyright Edward Tsang

Searching

Artificial Intelligence

≈ Knowledge representation + Search

Search Space = the set of all possible solutions under the given representation

Complete Search
Systematically explore every candidate solution in the search space

Incomplete Search
Use heuristics to search in promising areas for solutions

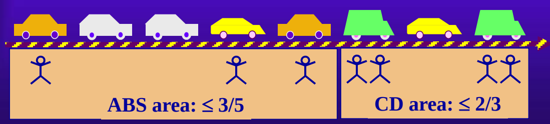
17 December 2010 All Rights Reserved, Edward Tsang

Stochastic Search

- Incomplete search
  - i.e. even if solutions exist, they may not be found
- Evolutionary computation
  - To evolve solutions thru maintaining a population
- Hill Climbing
  - To heuristically improve on the current solution
- Many more
  - Tabu search, guided local search, neural network, …

Car Sequencing Problem

Options					
ABS	×	√	√	×	
CD	×	×	√	√	
…					Total:
Production:	30	30	20	40	120



Combinatorial Explosion in Car Sequencing

- Schedule 30 cars:
  - Search space: 30 factorial ≅ 10^{32} leaf nodes
- Generously allow:
  - Explore one in every 10^{10} leaf nodes!
  - Examine 10^{10} nodes per second!
- Problem takes over **32 thousand years** to solve!!!
  - **$10^{32} \div 10^{10} \div 10^{10} \div 60 \div 60 \div 24 \div 365 \cong 31{,}710$**
- How to contain ***combinatorial explosion***?

Solution Space

- Suppose you have 2 variables, *x* and *y*
- Then you have a 2-dimensional solution space
- If the variables can take any real number value, then your solution space is continuous
- If *x* and *y* can only take integer variables, then your solution space is discrete
- If you have n variables, then you have an n-dimensional space

Search Space

- Sometimes interchanged with "solution space"
- The search method defines the paths that one can take in the solution space
- For example, the search space in branch and bound is a tree
- Good knowledge of the search space can sometimes help solving the problem
  - E.g. TSP, "2-opting", "basins of attraction"

Conclusion

- Problems are only hard if they have an exponential solution/search space
- Use specialized methods when available!
- Combinatorial Explosion haunts systematic search
- Stochastic Search is more practical
- There are many search methods
- Given a problem, which method to use?
  - Knowing your search space helps!