Algorithms and their Applications CS2004 (2020-2021)

Dr Stephen Swift

11.1 Applications, Introduction and Parameter Optimisation

Class Tests So Far...

- **Class Test CRI:** 161 attempts
- **Class Test CRII:** 67 attempts
- **Class Test CRIII:** 35 attempts
- **Remember to attain an E range grade, all four class tests must be passed!**
- **All class tests needs to be completed by 16/02/2021**

Previously On CS2004...

- So far we have looked at:
  - Concepts of Computation and Algorithms
  - Comparing algorithms
  - Some mathematical foundation
  - The Big-Oh notation
  - Computational Complexity
  - Data structures
  - Sorting Algorithms
  - Various graph traversal algorithms
  - Heuristic Search
  - Hill Climbing and Simulated Annealing

For the Rest of This Module...

- We are going to look into applications and advanced Heuristic Search methods
- We are going to cover:
  - Applications, introduction and parameter optimisation
    - This lecture!
  - Genetic Algorithms (method)
  - Evolutionary Computation (method)
  - Ant Colony Optimisation (method)
  - Particle Swarm Optimisation (method)
  - Bin Packing (application)
  - Data Clustering (application)
  - The Travelling Salesman Problem (application)

Applications – Part 1

- One of the aims of this module to provide a set of skills and techniques that can be applied to solving real world problems and applications
- However many applications require a high degree of previous knowledge and/or data
  - E.g. Gene expression data analysis
    - Very large volumes of data
    - Requires in depth molecular and biological knowledge

Applications – Part 2

- However many problems can be solved if a similar problem can be solved
- The **Scales** problem is easy to understand and requires only a little data
- If we have a method that can solve the Scales problem then we can solve the much more complex **Subset Selection** problem
  - We will look at examples of this later

Applications – Part 3

- Nearly all of the problems we will look at are **NP-Complete/NP-Hard**
- There is no algebraic or direct solution
- The computation complexity is exponential in terms of problem size
- E.g. The **Scales** problem is $O(2^n)$
- I.e. The applications we are going to look at are very hard to solve….

Optimisation – Definition

“The process of making something optimal”

- For the purposes of this module, the “thing” that we are making optimal is a solution to a problem
- We will often look for the highest or lowest value of a fitness function
- The elements that we change to seek an optimal solution are often called **parameters**, **variables** and/or **features**
  - We change the allocation of weights to optimise how well balanced a set of scales is using Hill Climbing to solve the Scales problem

Optimisation Problems

- We will be looking at the following different types of Optimisation problems:
  - Parameter Optimisation
  - Combinatorial Optimisation
    - Subset selection
    - Permutation problems
    - Grouping (Clustering) problems
    - Bin Packing
  - Multi Objective Problems
  - Constraint Satisfaction Problems

Parameter Optimisation – Part 1

- This is where we are searching for the values of a set of numbers that solve a problem, typically expressed as an equation
  - I.e. The maximisation or minimisation of a function
- More formally:

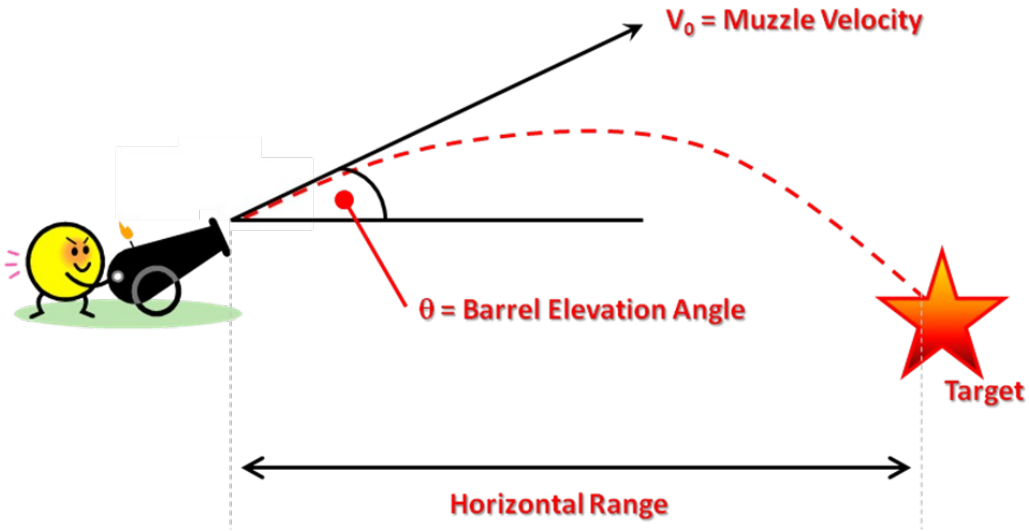
- Find the X that maximises/minimises $F(X)$ given:

$$X = (x_1, x_2, x_3, \ldots, x_n)$$

$$x_i \text{ is a number}$$

- Note that the exact definition of $F(X)$ is problem dependant

Parameter Optimisation – Part 2



- An example parameter optimisation problem is that of a very long range cannon
  - We have two parameters that need to be chosen so that we accurately hit a target
  - We cannot use the laws of motion since air density and drag need to be accounted for
- We can search for the optimal set of parameters
  - Hill Climbing
  - Simulated Annealing
  - Many, many more….
  - We will cover this topic in this weeks laboratory…

Combinatorial Optimisation

“The process of finding an optimal ordering or set from a number of objects or items”

Subset Selection – Part 1

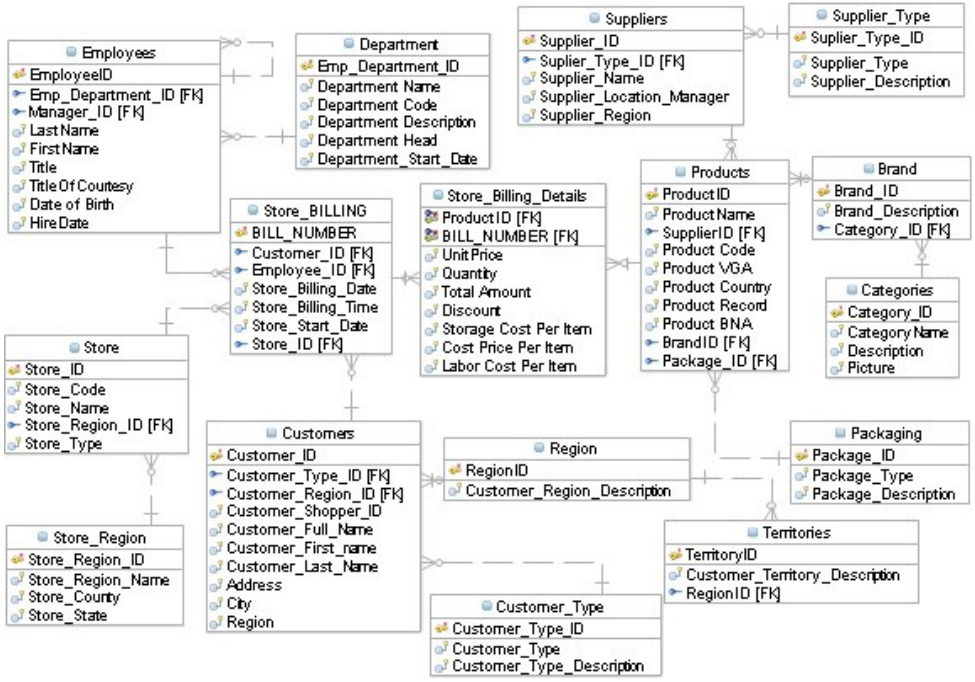
- Given a set of items, select a subset of the set that optimises some requirement
- More formally:

Given X, find a $Y \quad X$ such that $F(Y)$ is a maximum/minimum where X is a set of objects/items

- If you can solve the **Scales** problem then you can solve the subset selection problem

Subset Selection – Part 2

- A good example of Subset Selection is the **Feature Subset Selection** problem
- This is where we are trying to model a set of data, but have too many features/variables/Description
- We wish to select the best subset that models the problem
- Those of you who choose the **AI** option in level 3 will look into this problem:
  - Machine learning
  - Classification



Permutation Problems – Part 1

- A **permutation** is defined as a shuffling of a set of objects
  - A permutation of A, B, C, D is B, C, A, D
  - A permutation of **1, 2, 3, 4** is **4, 1, 2, 3**
  - If we have N objects then there are $N!$ possible permutations
  - $4! = 24,\ 10! = 3{,}628{,}800,\ 100! \approx 9.333 \times 10^{157}$

Permutation Problems – Part 2

- Without loss of generality we will assume that our solution is a permutation of the numbers

$$X = [1, 2, ..., N]$$

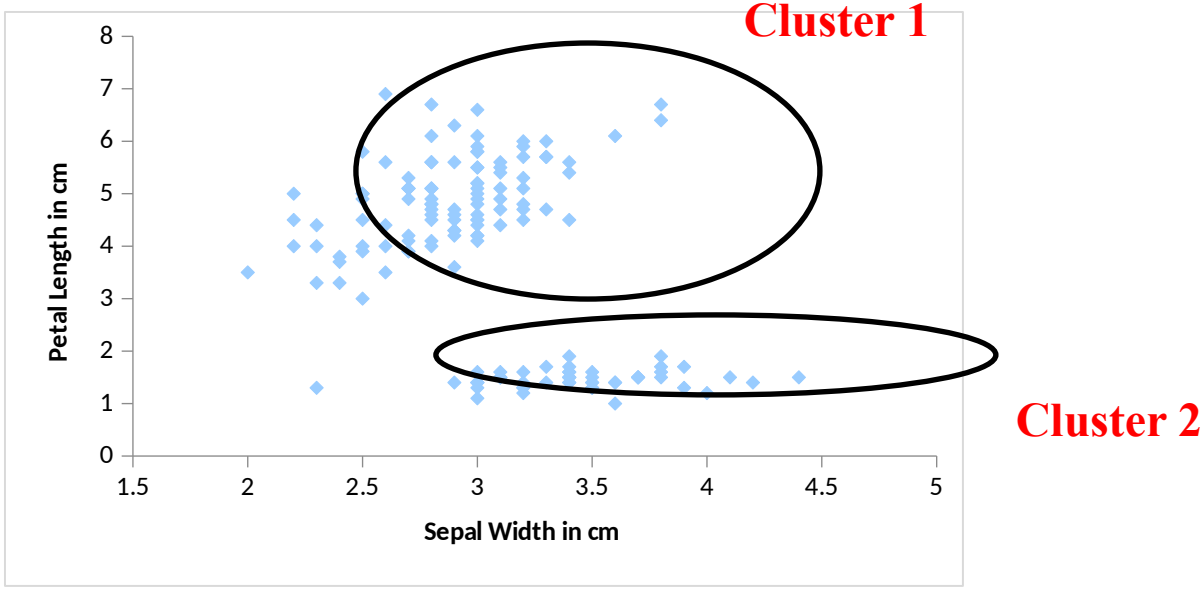
- We will use the notation $Y = \pi(X)$ to denote that Y is a permutation of X
- We formally define the permutation problem as find a $Y = \pi(X)$ such that $F(Y)$ is a maximum/minimum
- Note that the exact definition of $F(Y)$ is problem dependant

Permutation Problems – Part 3

- The **Travelling Salesman Problem (TSP)** is a classic example of a permutation problem
  - A sales person has to visit N cities as part of their job
  - The aim is to start off at one of the cities, visit each city exactly once and then to arrive back at the starting city
  - This is called a **tour**
  - The objective is to find a tour where the sum of the total distance travelled is a minimum
  - I.e. This is analogous to the sales person trying to minimise their petrol costs
- Examples include parcel delivery, general vehicle routing and road gritting
- We will cover this topic (TSP) in **much** more detail later
- We will look into the **Eight Queens** problem when we look at **Genetic Algorithms**

Data Clustering – Part 1

- Data Clustering is the process of arranging objects (as points) into a number of sets (k) according to "distance" or similarity
- Each set will be referred to as a cluster/group
- For the purposes of this module, each set is mutually exclusive, i.e. an item cannot be in more than one cluster

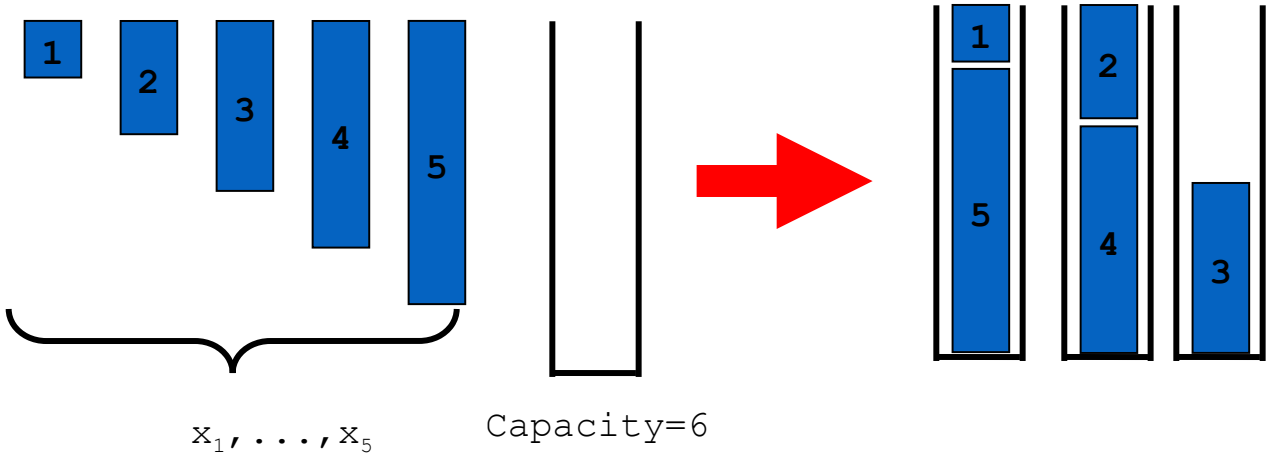


Data Clustering – Part 2

- Why Cluster?
  - Knowing which objects are highly related to other objects is very useful within data analysis
  - Less complex to model
  - May give insight into the unknown properties of some of the objects
  - A useful pre-processing tool
- The number of applications of data clustering is extensive
  - Gene Expression Data
  - Software Modularisation
  - Clustering Customer Behaviour
  - Etc…
- We will look into data clustering in **much** more detail (in the next lecture)

Bin Packing – Part 1

- The **bin packing** problem is where a number of n items of size $x_1, \ldots, x_n$ need putting into the smallest number of bins (or boxes) of size/capacity c



Bin Packing – Part 2

- There are a large number of bin packing applications:
  - 1-dimensional e.g. CD compilations
  - 2-dimensional e.g. stock cutting
  - 3-dimensional e.g. van packing
- Again, we will look into this in more detail next week…

Multi Objective Optimisation

- Some optimisation problems have more than one quality or fitness function
- Thus with these types of problems we might have more than one way of rating a solution
- For example we might be trying to optimise $X = (x_1, x_2, x_3, \ldots, x_n)$
- We might have a number of fitness function $F_1(X), F_2(X), \ldots, F_m(X)$
- The problem is how to compare two potential solutions when only some of the fitness function evaluations are better
- For example if we had two potential solutions X_1 and X_2 and three fitness functions such that:
  - $F_1(X_1) = 10, F_2(X_1) = 20$ and $F_3(X_1) = 30$
  - $F_1(X_2) = 20, F_2(X_2) = 10$ and $F_3(X_2) = 31$
- Which solution is better?

Weighted Fitness

- One solution to this problem is to assign a numerical weight or importance to the fitness functions
- For example:

- This assumes that we can assign a weight
- We must have some prior knowledge of how important each function is
- This also requires us to be aware of what the ranges of the fitness functions are
- For example if one function is always less than 1 and another is always greater than 1,000,000 then equal weights would be pointless

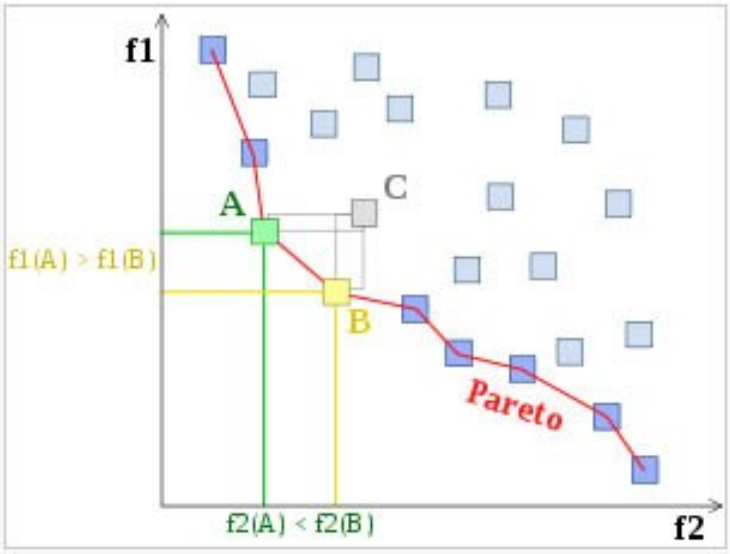
Pareto Optimality – Part 1

- Pareto optimality is a very straight forward but highly effective technique
- If we have two solutions X_1 and X_2 then we say that X_1 **Pareto Dominates** X_2 if $F_i(X_1)$ is better than $F_i(X_2)$ for all i
- We use the notation:

- If neither solution dominates each other then they are **Pareto Equivalent**
- Thus we could search for a solution to a problem which is not dominated by any other solution

Pareto Optimality – Part 2

- The **Pareto Front** is the set of **Pareto Equivalent** solutions that are not **dominated** by any other solution
- Multi Objective Optimisation has a number of applications:
  - Manufacturing
  - Telecommunications and computer networks
  - Logistics and transportation
  - Timetabling and scheduling



Constraint Satisfaction Problems

- Constraint Satisfaction Problems (CSP) are optimisation problems, where the valid solutions are constrained by a set of conditions
- For example, if we are seeking to optimise:

$$X = (x_1, x_2, x_3, \ldots, x_n)$$

- We might have a set of constraints such as

CSP Applications

Advanced Planning and Scheduling

- Train scheduling
- Production scheduling

Assignment problems

- Gate allocation for airports
- Balancing work among different people

Network management and configuration

- Planning of cabling of telecommunication networks
- Network reconfiguration without service interruptions

Database systems

- Ensure and/or restore data consistency

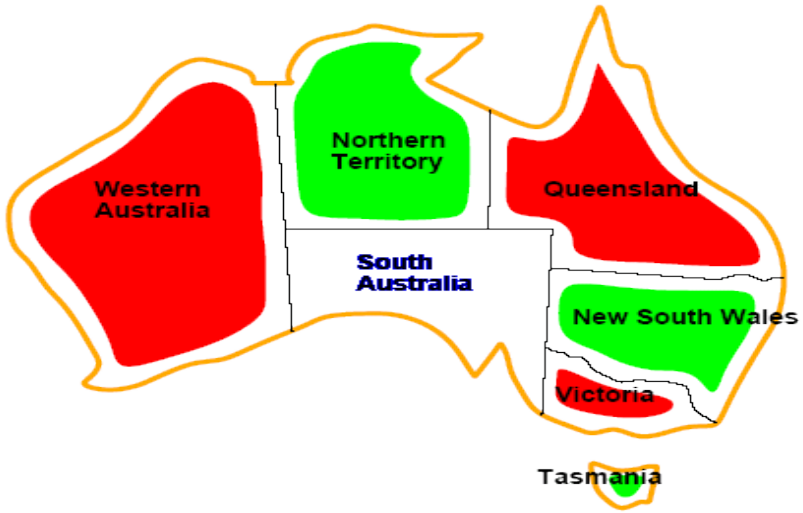
Molecular biology

- DNA sequencing
- Chemical hypothesis reasoning
- Protein docking

Electrical engineering

- Fault location
- Circuit layout computation

CSP Example



- Map Colouring
  - Colour all of the regions in a map so that no adjacent regions are the same colour
  - E.g. $x_i \neq x_j$ if region i is next to region j
  - Sometimes we wish to minimise the number of colours (4…)

CSP Strategies

- Often Heuristic Search methods are used to solve CSP
- General Strategy
  - Instantiate all variables randomly
  - Use a repair or update strategy to move towards a more and more complete solution
  - Stop if a complete solution is found
- Hill Climbing
  - Modify the value of one variable such that more constraints are satisfied
  - Restart with a random assignment if no more constraints can be satisfied (local minimum)
- Minimising conflicts
  - Randomly choose a variable that is involved with an unsatisfied constraint
  - Pick a value that reduces the number of unsatisfied constraints
  - If no such value exists pick a random value which does not increase the number of unsatisfied constraints

The Laboratory

- We will be looking at solving the “cannon” problem using a Hill Climbing Algorithm
- You can [**should**] re-use code from previous laboratories

Next Lecture

- We will be looking at Tabu Search and Iterated Local Search