Computing Concepts and Paradigms

Lecture 3
Algorithmic Methods

• A candidate algorithm is in itself a search in the search space of solutions.
• While automated ways exist to find a suitable algorithm for a problem, humans normally have to do the work.
• How do we go about to devise an algorithm?
• There are certain techniques that help algorithm design: Algorithmic Methods
Searches and Traversal

• Many times one has to traverse some structure to find some data or operate on some data.
• When lists and arrays are used, loops are usually used.
• When trees are used, recursion is more natural:
  – Depth First
  – Breadth First
Computational Geometry

• When dealing with shapes and graphs, we tend to use computational geometry.
• These are intuitive in that humans can solve them visually.
• Example: Maximum distance between two points in a polygon
Divide and Conquer

• One can split the problem into smaller chunks and solve for each chunk, using the result of each to solve bigger problem.
• This may make the problem easier to solve.
• Example: we used this strategy in recursively finding the maximum number in a list; merge sort.
Greedy Algorithms

• This involves a blind search, always choosing the best option.
• It does not always work, but is normally quite intuitive to come by.
• Example: Railroad contractor problem (minimal spanning tree)
• Example of failed greedy algorithm: Weary traveler (minimal path) in a directed acyclic graph