Reconnect '04
Experimental Algorithmics with a Focus on Branch and Bound for Discrete Optimization Problems

Cynthia Phillips, Sandia National Labs

Jeff Linderoth, Lehigh
Jonathan Berry, Lafayette

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Discrete (Combinatorial) Optimization

Informally, a Combinatorial Optimization problem is the minimization (maximization) of an objective function over a finite set of feasible solutions.

Examples from everyday life:

- Scheduling family activities (just finding a feasible solution)
  - Vehicle routing
- Packing a moving van or dishwasher
- Budgeting

Complexity

Most of the problems we really want to solve are NP-complete: Any algorithm that computes the **optimal** solution for **all** input instances is likely to take an unacceptable amount of time in the **worst case**.

But we still need to solve them in practice:

- Approximation
- Restricted input instances
- Average Case

Solving Combinatorial Optimization Problems in Practice

Goal: Do the best we can in solving a **particular instance**.

- That's what people really want to solve
- Structural insight from optimal solutions to small problems leads to
  - Better approximation algorithms (theory)
  - Better heuristics (practice)
  - Better solution of bigger problems

Experimental Algorithmics

- Rigorous computer experimentation
  - How well does an algorithm/implementation perform?
    - Time, Approximation quality
  - Algorithm comparisons
- Algorithmic questions arising during development of robust, efficient software
  - Cache issues
  - Generation of synthetic data
- Algorithm Engineering

Branch and Bound

Branch and Bound is an intelligent (enumerative) search procedure for discrete optimization problems.

$$\min_{x \in X} f(x)$$

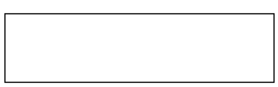
Three required (problem-specific) procedures:

- Compute a lower bound (assuming min) b(X)

$$\forall x \in X,\ b(X) \le f(x)$$

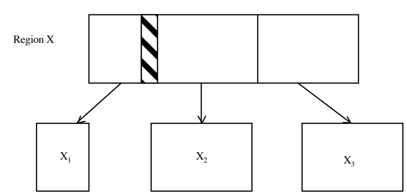
- Split a feasible region (e.g. over parameter/decision space)
- Find a candidate solution $x \in X$
  - Can fail
  - Require that it recognizes feasibility if X has only one point

Branching (Splitting)



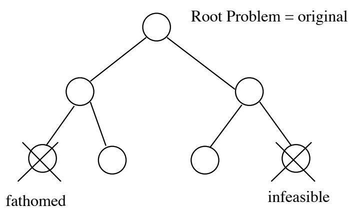
- Usually partitions the feasible region (or better)

Branching (Splitting)



- Usually partitions the feasible region (or better)

Branch and Bound



- Recursively divide feasible region, prune search when no optimal solution can be in the region.

Terminology

- The best feasible solution found so far is called the **incumbent.**
- A node awaiting bounding or splitting (not done) is called **active**.
- The set of active nodes is sometimes called the **frontier**.

Branch and Bound Solution Quality

- At any point in the computation, we have a global lower bound: the lowest lower bound among all open (active) nodes.
- We can stop at any point and compute an instance-specific approximation bound: value of incumbent/global bound
- If B&B runs to completion we have
  – Found an optimal solution
  – Proven that this solution is optimal

Example: Binary Knapsack

Given set of objects 1..n
Each has weight/size w_i, and value v_i

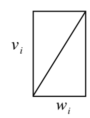
Thief wants the most valuable set that fits into a knapsack of size W

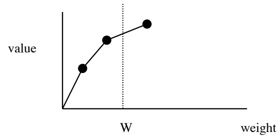
Find a subset of objects S that
Maximizes $V(S) = \sum_{i \in S} v_i$

Subject to $W(S) = \sum_{i \in S} w_i \leq W$

Binary Knapsack - Upper Bound (and Candidate)

- Bound with greedy strategy
  - Maximize value/unit weight
- Sort objects by "bang for the buck": $\frac{v_i}{w_i}$
  - Add objects in order
  - Split last object





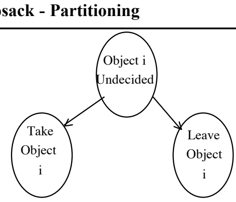
Greedy Isn't optimal

- Knapsack is weakly NP-complete

	w	v	v/w
A	6	12	2
B	5	9	9/5
C	4	4	1

- For W = 9, {B,C} is optimal

Knapsack - Partitioning



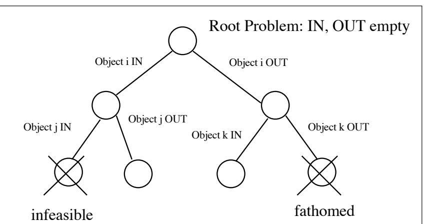
Represent a decision region with three subsets: IN, OUT, UNDECIDED

Upper bound and candidate: (value of) greedy on

UNDECIDED with new bound W - W(IN).

Branch object (i above): The one partially-included item in the greedy solution

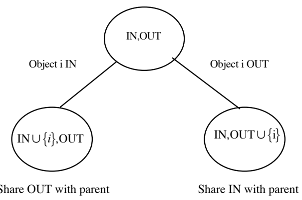
Branch and Bound



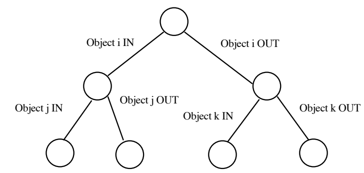
- Recursively divide feasible region, prune search when no optimal solution can be in the region.

Representation - Space issues

Two Lists (IN and OUT)
UNDECIDED is implied



Representation Issue - Space



- 3-valued vector: IOU
- Save difference with parent
- Issue: length of chains vs. space cost of replication

This course covers

- Practical ways to solve combinatorial optimization problems with branch and bound
- Integer Programming
  - Natural general way to express any NP-complete problem (most useful combinatorial optimization problems)
  - Theoretical foundations as relevant to practice
  - Software tools
- Experimental Algorithmics
  - Answering “so what?” for your experimental study

We’ll see example applications, but they’re the tip of the iceberg