# IE 495 Lecture 26

December 5, 2000

# Reading for This Lecture

- Primary
  - See me if you want references

# Example: Vertex Coloring

- Upper Bound:
  - Order nodes by degree
  - In order, color each node according to lowest feasible color
- Lower Bound: Maximum clique (in augmented graph)
- State Space: $x_i$ = color of $i^{th}$ vertex (1 ... k)
- Branching Rule: Branch on node with least number of feasible colors
- Search Strategy: It depends...

# Example: Traveling Salesman Problem

- IP Formulation

$$\begin{array}{llll} \max & c^T x_{ij} & & \\ & \Sigma_i \, x_{ij} & = & 2 \\ & \Sigma_{\,(i,j)\in \delta(S)} x_{ij} & \geq & 2 \\ & x_{ij} \text{ integer} & & \end{array}$$

- Variables correspond to the edges of the graph

# Generating Constraints

- Dynamically generate violated constraints.
- Find the minimum cut in the *fractional graph*.
- If minimum cut is < 2, then we have a violated constraint.
- Otherwise, there are no violated constraints.

# Generating Columns

- Columns correspond to edges.
- Keep track of constraints currently in the LP relaxation.
- Columns can be constructed dynamically by examining the constraints already in the LP.

# Basic Branch and Cut

- Start with only a small set of columns and the degree constraints in the LP relaxation.
- Solve the LP dynamically (generating constraints) until no further improvement is possible.
- Branch on a fractional variable.
- Continue solving nodes dynamically in this fashion.
- Before pruning, check to see whether any of the columns not in the LP could enter the basis.
- If so, add them and continue.

# Constraint and Column Pools

- To avoid generating the same objects in different parts of the tree, keep track of the "best" in a global pool.
- Access the pool to see if anything is available before generating something new.
- This is very efficient for problems where constraint and/or column generation is expensive.

# Course Wrap-up

# What will this class be about?

- Some computer science theory
  - data structures
  - design and analysis of algorithms, complexity
- Some mathematical theory
  - matrix computations, linear algebra
  - recursion, induction
- Some programming/development
- The ultimate focus will be on bringing all these tools together to solve problems.

# Topic Coverage

- Fundamentals of Computer Systems
- Models of Computation/Complexity theory
- Induction and Recursion
- Parallel Programming/Parallel Algorithms
- Basic Data Structures
- Basic Algorithms
- Numerical Analysis
- Advanced Algorithms

# Conclusions

- It's been a pleasure teaching the class.
- You have all come a long way since the beginning.
- I have pushed your limits and you have responded.
- I hope you have a deeper appreciation of what it takes to implement the techniques of operations research.
- If you are interested in more study of any of the topics we covered, please talk to me.

The End