MAE 552 – Heuristic Optimization

Lecture 26

April 1, 2002

Topic:Branch and Bound

Branch and Bound

We have seen this semester that the size of real-world problems grows very large as the number of design variables increases.

Recall that there are (n-1)!/2 different solutions for the Travelling Salesman Problem (TSP).

Exhaustive search is impractical when n>20

It would be helpful of we could reduce the size of the search space where we know the optimum solution will nc exist.

Branch and Bound

Branch and Bound works on the idea of successively partitioning the design space.

1st we need some means on determining a lower bound on the cost of any particular solution.

A *lower bound* on a solution means the solution will cost at least the value of this lower bound.

If we are maximizing the we need to find an *upper bound* on a solution - *a value which this solution cannot exceed*

Branch and Bound

For minimization

- If we have a solution 1 with a cost *c*
- **AND** we know that another solution 2 has lower bound that is greater than *c*
- **THEN** we do not need to evaluate 2 because we know that 2 will exceed 1.

Branch and Bound

For maximization

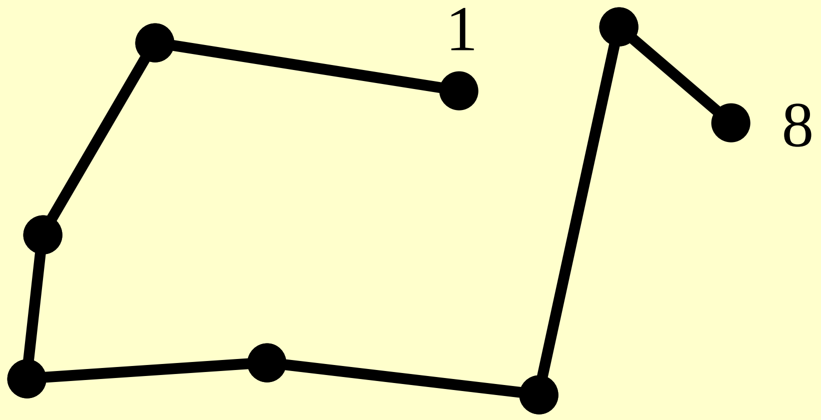
- If we have a solution 1 with a cost *c*
- **AND** we know that another solution 2 has upper bound that is less than *c*
- **THEN** we do not need to evaluate 2 because we know that 2 will never exceed 1.

Branch and Bound

We can determine an lower or upper bound by partially evaluating a particular solution.

Example using TSP:

Say we evaluate a partial tour of a TSP with 15 cities and after 8 cities it already exceeds our best solution so far.



here is no need to evaluate the rest of the tour

ND there is no need to evaluate the other tours that start wi
ose 8 cities!!!!

Branch and Bound

The design space can be organized in a tree structure

The branch and bound prunes away branches that are not of interest.

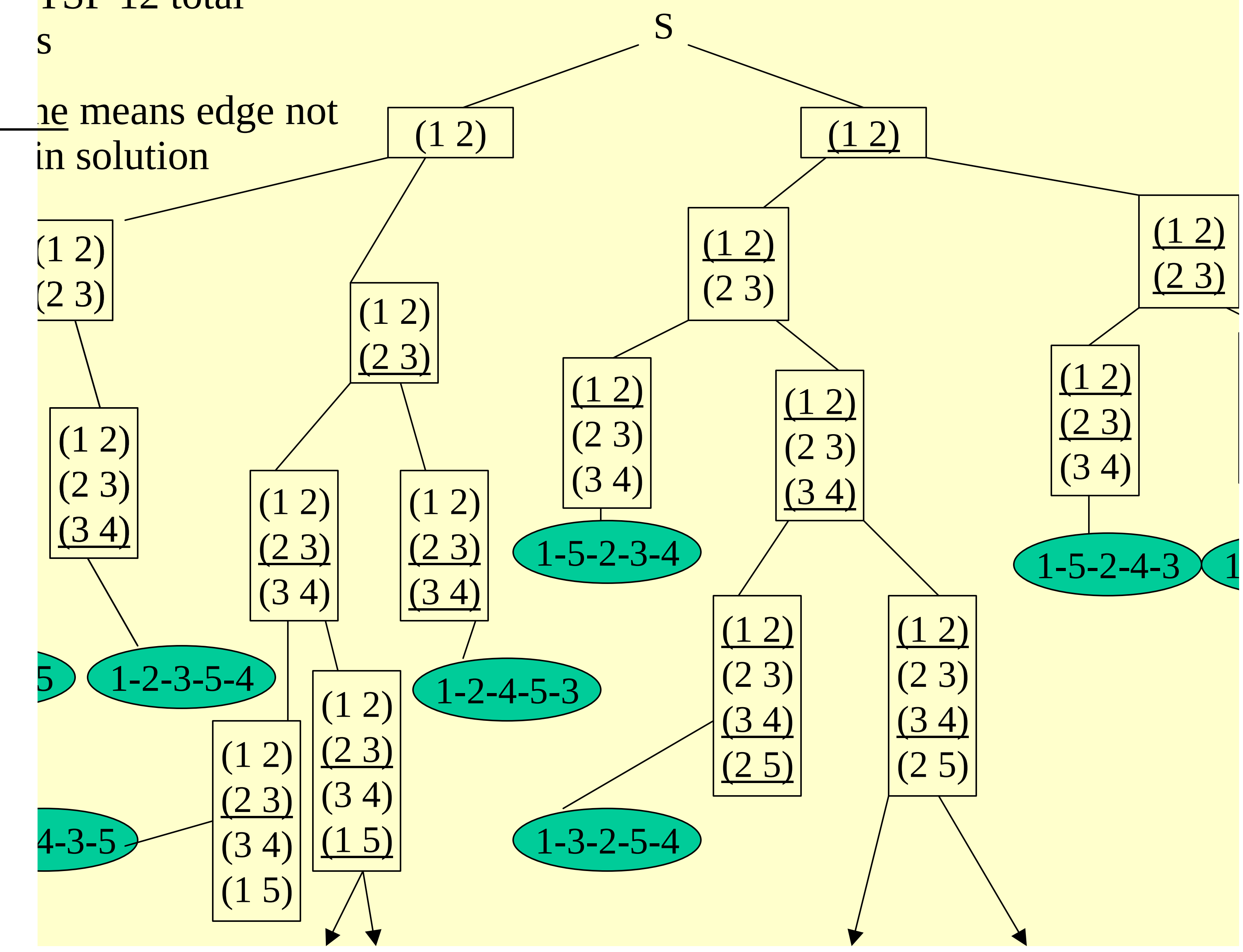
The design space of the TSP can be organized on the basis of whether or not edge (1 2) occurs in the solution.

It can be further divided into branches where edge (2 3) appears and so forth.

Branch and Bound

r the search space of
TSP 12 total
s

ne means edge not
in solution



Branch and Bound

Suppose the cost can for travelling between cities is described by the following cost matrix.

C =

0,7,12,8,11
7,0,10,7,13
12,10,0,9,12
8 ,7 ,9 ,0 ,10
11,13,12,10,0

•Each entry is the cost of travelling from a city in the ith ro
to one in the jth column.

•The zeros down the diagonal indicate that you cannot trav
from a city to itself

Branch and Bound

Given the search tree we need a heuristic for estimation a lower bound on the cost of any final solution, or even any solution containing a particular node (i.e. city)

If the lower bound is higher than the best solution we have found so far, we can keep looking without having to actually compute its final cost.

Branch and Bound

Here is a simple but not very effectual way to compute a lower bound for a tour.

Consider a complete solution for the TSP.

Every tour comprises 2 adjacent edges for every city, one edge enters the city, one edge goes on to the next city.

If we select the two shortest edges that are connected to each city and take the sum of these edges divided by 2 we will obtain a lower bound.

We could not possibly do better because this selects the very best edges for all the cities.

Branch and Bound

With respect to the cost matrix this turns out to be

$$C = \boxed{\begin{matrix} 0,7,12,8,11 \\ 7,0,10,7,13 \\ 12,10,0,9,12 \\ 8\ ,7\ ,9\ ,0\ ,10 \\ 11,13,12,10,0 \end{matrix}}$$

$$[(7+8)+(7+7)+(9+10)+(7+8)+(10+11)]/2=84/2=42$$

ote: 7 and 8 correspond to the two best values in column 1

- Note that 7 and 8 in the first parentheses correspond to th lengths of the two shortest edges connected to city 1 whereas the 7 and 7 correspond to the lengths of the two shortest edges connected to city 2 and so on

Branch and Bound

At first glance this may seem to be good way find a good solution but it is important to note that determining the lower bound does not specify a solution.

It is not possible to specify a solution that incorporates all of these shortest edges because it is generally necessary to specify bad edges to form a legal tour.

Branch and Bound

Once some edges are specified we can incorporate that information and calculate a lower bound on that partial solution.

If we knew that edge **(2 3)** were included but edge (1 2) was not then the lower bound on the partial solution woul be:

(8+11)+(7+10)+(9+10)+

7+8)+(10+11)]/2=45.5

C =

0,**7**,12,8,11
7,0,**10**,7,13
12,10,0,9,1
8 ,7 ,9 ,0 ,1
11,13,12,10,

Branch and Bound

We can improve the lower bound by including the implied edges or excluding those that cannot occur.

If we determined that edges (1 2) and (2 4) were included in a tour then we would get a lower bound of 42.

But with these two edges included we can exclude edge (1 4) which would raise the lower bound to 44.

Branch and Bound

Exercise: Show that with edges (1 2) and (2 4) included the lower bound is 44.

$$C = \boxed{\begin{array}{c} 0,7,12,8,11 \\ 7,0,10,7,13 \\ 12,10,0,9,12 \\ 8\ ,7\ ,9\ ,0\ ,10 \\ 11,13,12,10,0 \end{array}} \quad [(\quad)+(\quad)+(\quad)+(\quad)+(\quad)]/2$$

Branch and Bound

It is important to recognize that is cost time to compute the lower bounds

The cost of computing the lower bounds has to be made up in the time saved in pruning the tree.

So we want the best lower bound possible, to ensure efficient pruning.

This is the subject of much research.

Branch and Bound - Continuous Problems

The branch and bound algorithm can also be applied to continuous problems.

The idea here is to iteratively subdivide the design space into regions and check each region to see if there is a singl or multiple local optima.

This check is performed by testing to see if the partial derivatives are always negative or positive.

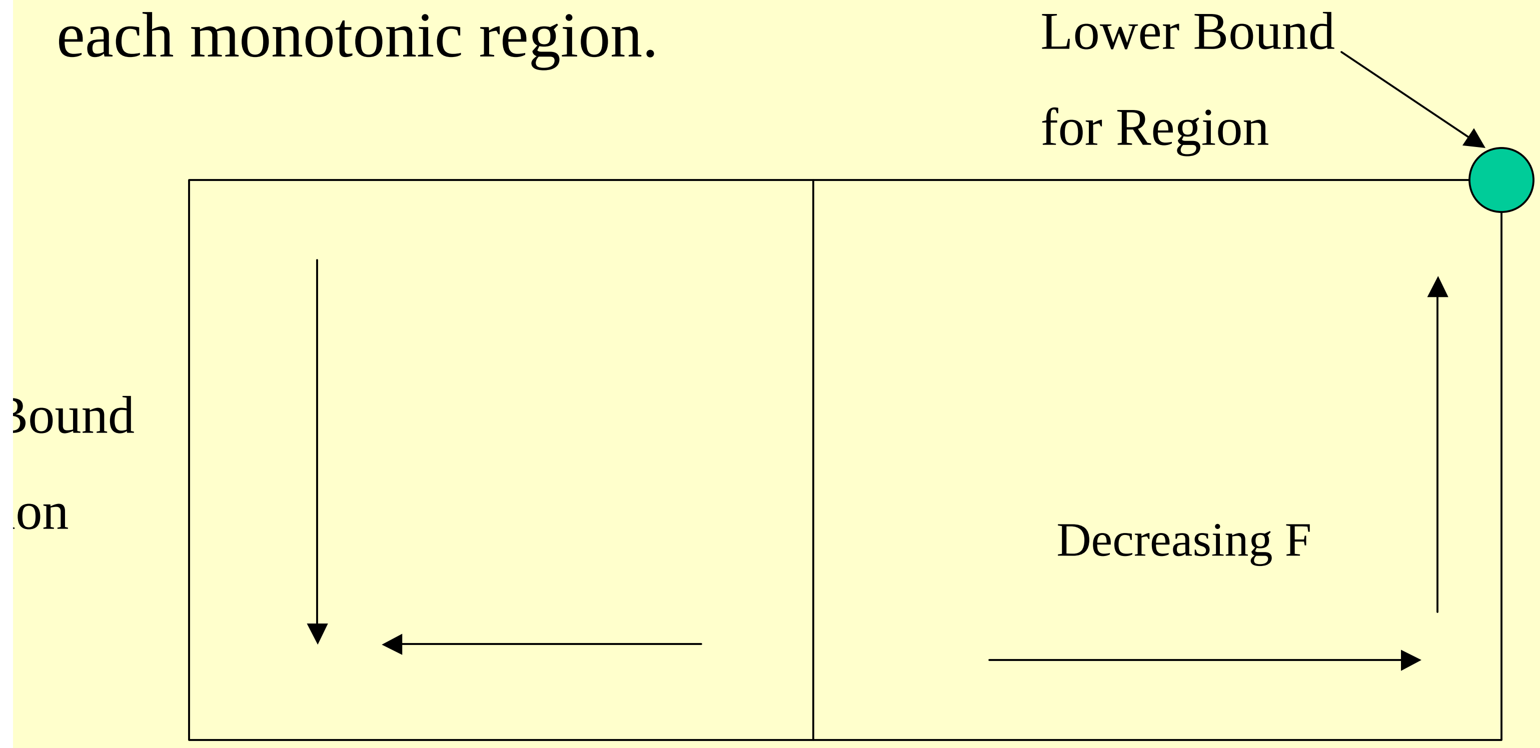
If the area is monotonic then a lower bound is computed fo it.

If not, then the region is further subdivided.

Regions are eliminated whose lower bounds are greater than the best solution found thus far.

Branch and Bound

Lower bounds can be computed by looking at the edges of each monotonic region.



Branch and Bound

There are many variants of this generic branch and bound algorithm

- Interval analysis can be used to determine the bounds where the calculations are performed on intervals instead of on real number
- A stochastic version of the algorithm calculates f at random points to determine a lower bound.

We will go into the application of Branch and Bound to continuous problems in more detail next lecture