CSE 6311
Advanced Computational Models and Algorithms

Instructor: Dr Gautam Das
Lecture notes for March 5, 2009

Compiled by Avinash Narayan
OUTLINE

- Euler Circuit
- PTAS for TSP
- Set Cover Approximation
*Euler circuit does not exist for a graph with one or more odd degree nodes

*Euler Circuit Exists for a graph only if every node is of even degree

**Fig-1**

**Fig-2**
FLEURYS ALGORITHM

Finding Euler Circuit in a graph of even degree at every node.
1. Pick any vertex as a starting point.
2. Marking your path as you move from vertex to vertex, travel along any edges you wish.
3. If you find a cycle, separate the vertices in this cycle.
4. Find all cycles in graph.
5. Continue until you return to your starting point.
6. Patch the cycles.

For a traversal of graph in Fig-2 along path a,b,c,d,e,f,c,g,b,i,h,g,i a we find the above cycles.
PTAS FOR TSP

Polynomial Time Approximation Schemes break through the hard barrier in Approximation ratios. These scheme design solutions to a NP complete problem as close as we like it to the optimal solution but the tradeoff is the running time.

For a given constant value “$\varepsilon$” \( C_{\text{approx}} \leq (1 + \varepsilon) C_{\text{opt}} \)

For a TSP problem, If we assume Distances are Geometric/Euclidean/Vector based, we can achieve better Poly. Time Approx. ratios lower than \( 3/2 \) ratio(Christofide’s Algo)

Running Time for TSP = \( O(n \log n^{O(1/\varepsilon)}) \)  


“The idea is to snap the nodes(greater than grid points) onto nearby grid points and apply algorithm on this grid problem”

For a given value of $\varepsilon = 0.001$, It is still in order of \( n(\log n)^{1000} \)
The greedy method works by picking, at each stage, the set $S$ that covers the greatest number of remaining elements that are uncovered.

$$S_{\text{approx}} = \{ S_1, S_4, S_5, S_6/S_3 \}$$

We need to prove that the approximation factor is always $\log(n)$

(where “n” is the number of elements in the set)
AMORTIZED ANALYSIS

\[ X = \{1, 2, \ldots, i, \ldots, n\} \]

\[ X_i \] [Mass associated with \(i^{th}\) Element]

\[ \sum_{i=1}^{n} x_i = C_{\text{approx}} \]

\[ \sum_{s \in S_{\text{opt}}} \sum_{x \in s} x_i \geq C_{\text{approx}} \]

\[ C_{\text{opt}} \max_{s \in S} \left\{ \sum_{x \in s} x_i \right\} \geq C_{\text{approx}} \]
The intuition is if this set itself was the maximum set then our greedy algo would have chosen it

\[ C_{\text{opt}} \sum_{i=1}^{C_{\text{approx}}} (U_{i-1} - U_i) \cdot \frac{1}{U_{i-1}} \geq C_{\text{approx}} \]

\[ C_{\text{opt}} \sum_{i=1}^{n} \frac{1}{i} \geq C_{\text{approx}} \]

\[ C_{\text{opt}} \int_{1}^{n} \frac{1}{x} \, dx = n \ln(n) = nO(\log n) \geq C_{\text{approx}} \]

So the ratio is always order of \( \log(n) \)