# Absolute Approximation

Maximum Program Stored

Edge Coloring

Other problems



# MINIMUM EDGE COLORING

- INSTANCE: Graph $G=(V,E)$.
- SOLUTION: A coloring of $E$, that is, a function $f$ such that, for any pair of edges $e_1$ and $e_2$ that share a common endpoint, $f(e_1) \neq f(e_2)$.
- MEASURE: Number of colors, i.e., cardinality of the range of $f$.
- **Comment**: Denote the maximum vertex degree by $\Delta$, then no coloring can use less than $\Delta$ colors. Vizing 1964 proved that it is possible to color the edges using no more than $\Delta+1$ colors. His proof has been translated to an $O(mn)$ time approximation algorithm that solves the problem within 1 off the optimum.

  On the other hand, Holyer in 1980 proved that identifying the chromatic number for a graph is NP-complete.



# Vizing's algorithm

**begin**
 $\Delta$ :=maximum degree of $G$;
 $G'$:=($V$, $E'$=ø); // $G'$ is clearly colorable with $\Delta$+1 colors
 **repeat**
  add an edge $(u,v)$ of $E$ to $E'$;
  extend coloring of $G'$ without $(u,v)$ into coloring of $G'$ with at most $\Delta$+1 colors;
  $E$ := $E$-{$(u,v)$};
 **until** $E$:=ø
**end.**

- To justify that the above is a polynomial-time algorithm to color a graph with at most $\Delta+1\leq$OPT(I)+1 colors, we need only to prove "the ($\Delta$+1)-coloring of $G'$ without $(u,v)$ can be extended into coloring of $G'$ with at most $\Delta$+1 colors in ploynomial time".

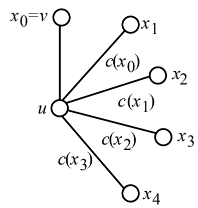


# Proof

- Assume $G'$ without $(u,v)$ has an edge-coloring with at most $\Delta$+1 colors
- Let $\mu(v)$ denote the set of colors that are not used to color an edge incident to $v$
- Clearly, if the coloring uses $\Delta$+1 colors, then for any $v$, $\mu(v) \neq$ø: let $c(v)$ be one of the colors in $\mu(v)$

## Proof (continued)

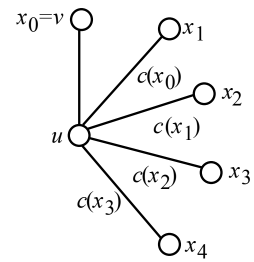
- Compute in polynomial-time a sequence of edges $(u,x[0]),\ldots,(u,x[s])$ such that:
  - $x[0]=v$
  - for any $i$, the color of $(u,x[i]) = c(x[i\text{-}1])$
  - there is no other edge $(u,w)$ such that its color is equal to $c(x[s])$

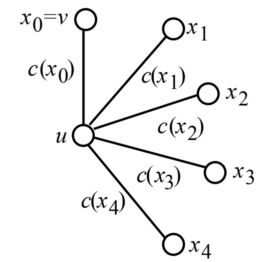





## First case: $c(x[s])$ is in $\mu(u)$

- In this case, we can simply shift the colors of the sequence in order to obtain the new coloring of $G'$
  - That is, for any $i$, we color $(u,x[i])$ with $c(x[i])$

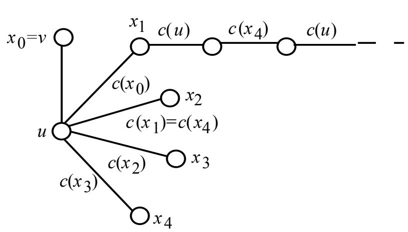





## Second case: $c(x[s])$ is not in $\mu(u)$

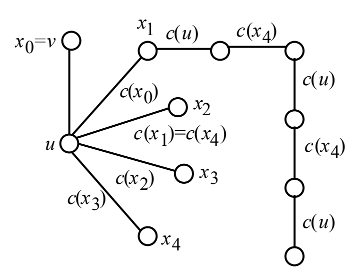
- In this case, one edge $(u,x[i])$ must have been colored with $c(x[s])$ (since the sequence is maximal)
  - Hence, $c(x[i\text{-}1]) = c(x[s])$
  - We compute in polynomial time a path $P_{i\text{-}1}$ starting from $x[i\text{-}1]$ formed by edges whose colors are, alternatively, $c(u)$ and $c(x[s])$

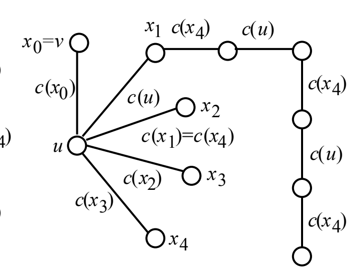





## First subcase: $P_{i\text{-}1}$ does not end in $u$

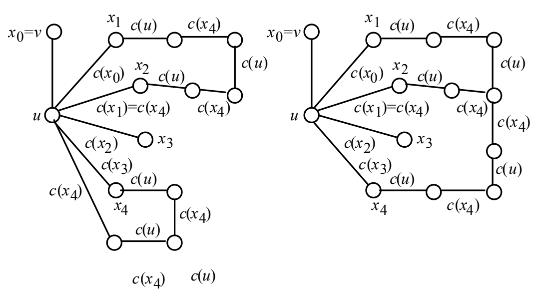
- Interchange colors $c(u)$ and $c(x[s])$ in the path, assign color $c(u)$ to $(u,x[i\text{-}1])$, shift the colors of the subsequence of edges preceding $(u,x[i\text{-}1])$



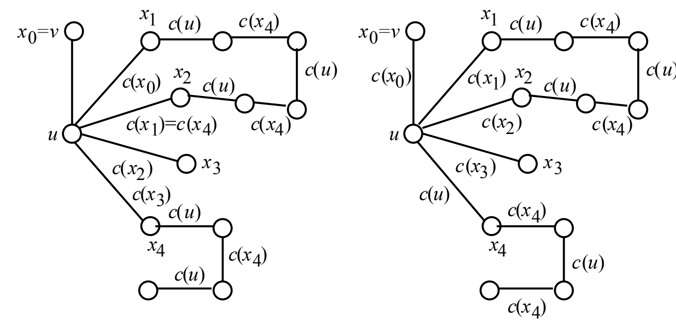
## Second subcase: $P_{i-1}$ ends in $u$

- Compute in polynomial time a path $P_s$ starting from $x[s]$ formed by edges whose colors are, alternatively, $c(u)$ and $c(x[s])$
  - $P_s$ does not end in $u$





- Interchange colors $c(u)$ and $c(x[s])$ in $P_s$, assign color $c(u)$ to $(u,x[s])$, and shift the colors of the subsequence of edges preceding $(u,x[s])$





## Other absolutely approximable problems

**There are a few other natural optimization problems for which approximation algorithms with a small additive error are known. Here are two examples.**

**Minimum Coloring of Planar Graph**

**Instance**: A planar graph $G=(V,E)$.

**Objective**: color the vertices of $G$ with a minimum number of colors so that any two vertices joined by an edge receives different colors.

**Comment**: Garey, Johnson, and Stockmeyer in 1976 has proved that it is NP-complete to decide if a planar graph is 3- colorable.

It is trivial to decide whether or not a graph is 2-colorable. This happens if and only if the graph is bipartite, a property can be verified in linear time.

The Four-color algorithm by Appek and Haken has received a great deal of attention. The algorithm colors any planar graph with at most 4 colors in polynomial time. It is hence a polynomial approximation algorithm within 1 unit off the optimum.

**Minimum-Degree Spanning Tree**

**Instance**: A graph $G=(V,E)$.

**Objective**: find a spanning tree of $G$ whose maximum vertex degree is minimized.

**Comment**: Any procedure for finding a minimum-degree tree in a graph could be used to identify whether the graph contains a Hamiltonian path. Thus the problem is NP-hard.

Fürer and Raghavacheri in 1994 devised a polynomial algorithm that approximates the problem within one unit off the optimum. This is an exciting discovery in 90's of the last century.



## Relative Approximation Algorithms

- 2-approximation algorithm for VC
- 2-approximation algorithm for Bin Pack
- 1.5-approximation algorithm for Metric TSP

## 2-approximation algorithm for Minimum Vertex Cover

- The algorithm:
  $C=\emptyset$;
  **for** any edge $(u,v)$ **do**
  **if** ($u$ is not in $C$) **and** ($v$ is not in $C$)
  **then** insert $u$ and $v$ in $C$;
  **return** $C$
- Feasibility: Obvious.
- Time: $O(|E|)$.
- Performance ratio: A(I)/OPT(I)≤2.

- Proof:
  - Let $M$ be the set of edges the algorithm takes their endpoints into $C$.
  - The edges in $M$ are mutually independent, and $|C|=2|M|$.
  - For each edge $(u,v)$ in $M$, either $u$ or $v$ must be included in a vertex cover.
  - Thus any optimal vertex cover $C^*$ must has $|C^*|\geq|M|=|C|/2$.



## Minimum Bin Pack

**Instance**: Finite set $I$ of rational numbers $\{a_1,\ldots,a_n\}$ with $a_i \in (0,1]$ .
**Solution**: Partition $\{B_1,\ldots,B_k\}$ of $I$ into $k$ bins such that the sum of the numbers in each bin is at most 1.
**Measure**: Cardinality of the partition, i.e., $k$.

- Next Fit (NF) algorithm:
  **for** each number $a$, **if** $a$ fits into the last open bin **then** assign $a$ to this bin **else** open new bin and assign $a$ to this bin.
- Time: $O(|n|)$.
- Performance ratio: $R_{NF}$=NF(I)/OPT(I)≤2.
- Proof:
  - Number of bins used by the algorithm is at most 2$A$, where $A$ is the sum of all numbers
    - For each pair of consecutive bins, the sum of the number included in these two bins is greater than 1
  - Each feasible solution uses at least $A$ bins
    - Best case each bin is full (i.e., the sum of its numbers is 1)
  - Performance ratio is at most 2
- There are better algorithms.
  - First Fit (FF) improves the performance ratio to 1.7
  - First Fit Decreasing (FFD) reaches a performance ratio of 11/9.



## Minimum Metric TSP

**Instance**: A complete graph $G=(V, E)$, A cost function $d$: $E\rightarrow Z^+$ satisfying the triangle inequality $d(u,v)+d(v,w)\geq d(u,w)$.
**Solution**: A Hamiltonian circuit $C$ in $G$.
**Measure**: The cost of $C$, i.e., $\sum_{e\in C} d(e)$.

- Christodes' algorithm (A):
  1. Compute the minimum spanning tree $T$ of the graph $G = (V, E)$. $O(|E|\log|E|)$ time
  2. Let $Q$ be the odd degree vertices in $T$. $|Q|$ is even. $O(|E|)$ time
  3. Compute a minimum cost perfect matching $M$ on the graph induced by $Q$. $O(|Q|^3)$ time
  4. Add the edges in $M$ to $E$. Now the degree of every vertex of $G$ is even. Therefore $G$ has an Eulerian tour. Trace the tour, and take shortcuts when the same vertex is reached twice. This cannot increase the cost since the triangle inequality holds. $O(|E|)$ time
- Time: $O(n^3)$.
- Performance: $A$(I)≤1.5$OPT$(I).

- Proof: let $C^*$ be the optimal solution (HC). We have:
  - $d(C) \leq d(C)+d(M)$
  - $d(T)/ d(C^*)<1$
    - since if we delete an edge of the optimal HC, a spanning tree results.
  - $d(M)/ d(C^*)\leq 1/2$
    - Consider the optimal HC $C'$ visiting only the vertices in $O$.
    - By the triangle inequality $d(C') \leq d(T)$.
    - $C'$ defines two disjoint matchings on the graph induced by $O$. At least one of these has cost of no more than $d(C')/2 \leq d(C^*)/2$.
  - Thus, $A$(I)/$OPT$(I)= $d(C)/ d(C^*)$ <3/2.