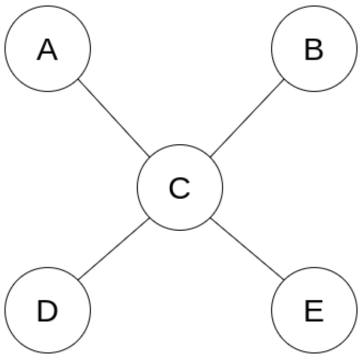
Advanced Algorithms and Data Structures
Assignment 5

Isabella Odorico `jkd427` Caroline Kierkegaard `qlj556` Mikkel Willén `bmq419` Fie Hammer `gsr530`

January 3, 2024

35.1-1

A graph like the following would result in a suboptimal solution everytime, since we pick an



edge, and add, the two verteces attached to the edge, to C, resulting in $|C| = 2$ where the optimal solution would be just the vertex c.

35.1-4

We create a greedy algorithm that finds the optimal vertex cover for a tree in linear time. If we make an algorithm, that finds the node with the most children, then add that, and remove the parent and the children from consideration. Continue doing this untill there are no more nodes to be considered.

35.1-5

No, we don't think the relationship implies that there is a polynomial-time approximation algorithm with a constant approximation ratio for the clique problem. We have made an example to justify our answer:

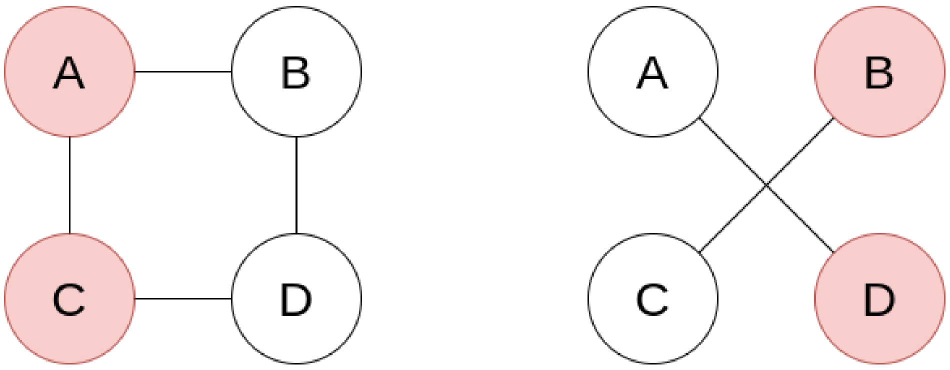


Figure 1: On the left hand side is an approximate vertex cover. On the right hand side is the complement to this vertex cover in the complement graph

The complement to the vertex cover in the complement graph, is not a maximum-size clique. It is actually not a clique at all. Thus, there are no constant ratio for which it is smaller than the maximum-size clique.

35.2-5

For a given crossing in the plane, we can reduce the problem to the 4 points spanning the cross. 2 of the points, not connected by the crossing line, will be connected at some other point in the graph G', so we can make a connection between these to points, with the sum of the lines connection them. See the below example

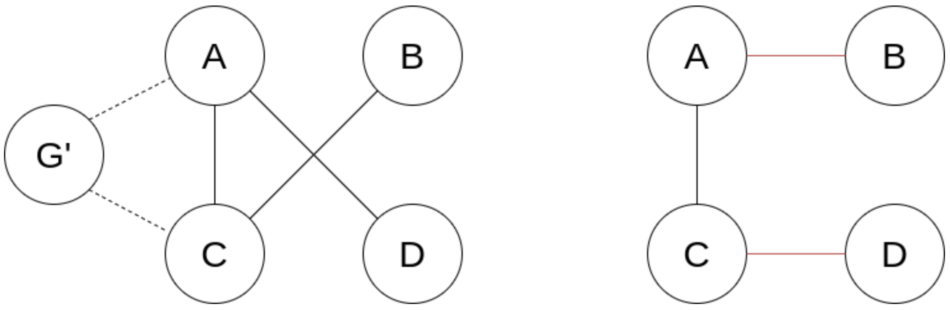


Figure 2: Caption

The distance of the sum of the diagonals will always be greater or equal to the distance of the sum of two sides of a square. Thus, by altering the graph as described above, the distance the salesman has to travel will be equal or smaller than before. There does not exist a shorter path than the optimal tour, so this implies that an optimal tour cannot cross itself.

35.3-3

We would implement the algorithm in roughly the same way, but when we have chosen the biggest subset, we then remove the points in the subset from the remaing subsets, and then take the next largest subset and repeat, untill the subsets have size 0.

35.4-2

This is our algorithm for the MAX-CNF satisfiability problem.

```
RANDOM-ASSIGNMENT(phi)
    for each variable x_i of phi
    choose x_i in {0, 1} by flipping fair coin
return assignment
```

It is a randomized 2-approximation algorithm. Proof: The worst case is if all clauses have 1 literal, because this would give the least posssible chance of satisfiability. Let $\Phi = C_1 \wedge \ldots \wedge C_n$. Consider $C_i = l_1$. Clause C_i not satisfied if $\neg l_1$, which gives us the probability

$$\Pr[\neg C_i] = \Pr[\neg l_1] = \frac{1}{2}$$

which means

$$\Pr[C_i] = 1 - \Pr[\neg C_i] = 1 - \frac{1}{2} = \frac{1}{2}$$

The number of satisfied clauses is

$$X = \sum_{i=1}^{n} [C_i]$$

We have the expectation

$$\mathrm{E}[X] = \mathrm{E}\left[\sum_{i=1}^{n} [C_i]\right] = \sum_{i=1}^{n} \mathrm{E}[C_i] = \sum_{i=1}^{n} \frac{1}{2} = n\frac{1}{2}$$

and we have the approximation ratio

$$\frac{C^*}{C} = \frac{C^*}{n\frac{1}{2}} \leq \frac{n}{n\frac{1}{2}} = 2$$

35.4-3

We write out the algorithm.

```
APPROX-MAX-CUT(G)
    for each v in V
        if rand(1.0) > 0.5
            add v to S
        else
            add v to V - S
```

This algorithm runs in linear time. If we for each edge $(u, v) \in E$ define the event $A_{u,v}$ to be the event where edge (i, j) crosses the cut $(S, V - S)$, and let $1_{A_{u,v}}$ be the indicator random varible for $A_{u,v}$.

The event $A_{i,j}$ occurs if and only if the vertices u and v are placed in different sets during the main loop in `APPROX-MAX-CUT`, and therefore we have

$$\begin{aligned} P[A_{u,v}] &= P[u \in S \land v \in V - S] + P[u \in V - S \land v \in S] \\ &= \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} \\ &= \frac{1}{2} \end{aligned}$$

We then let `opt` denote the cost of a maximum cut in G, and let $c = |(S, V - S)|$, which is the size of the cut produced by `APPROX-MAX-CUT`, then we have

$$c = \sum_{(u,v) \in E} 1_{A_{u,v}}$$

and

$$\texttt{opt} \leq |E|$$

which means we have

$$\begin{aligned} E[c] &= E\left[\sum_{(u,v) \in E} 1_{A_{u,v}}\right] \\ &= \sum_{(u,v) \in E} \mathrm{E}\left[1_{A_{u,v}}\right] \\ &= \sum_{(u,v) \in E} P\left[A_{u,v}\right] \\ &= \frac{1}{2}|E| \geq \frac{1}{2}\texttt{opt} \end{aligned}$$

and we have

$$E\left[\frac{\texttt{opt}}{c}\right] \leq \frac{|E|}{\frac{1}{2}|E|} = 2$$

and so we have that `APPROX-MAX-CUT` is a randomized 2-approximation algorithm.

35.5-2

We don't know where to begin, or how to solve this:(

Dispositions

Dispositions for Approximation Algorithms Isabella (jkd427)

- Introduction to approximation algorithms
  - What are approximation algorithms?

  - Why are approximation algorithms interesting?
- Approximation Ratio
  - Explanation and definition of approximation ratio
  - The general form of the approximation ratio ($max\left(\frac{C}{C*}, \frac{C*}{C}\right) \leq p(n)$)
- Vertex Cover
  - Vertex-Cover explanation
  - Approx-Vertex-Cover, example and proof (if time)
- 3-SAT
  - 3-SAT explanation
  - Max-3-SAT, example and proof (if time)

Caroline

- Why are approximation algorithms interesting?
- Definition of approximation ratio
- Example and proof of vertex cover
- Example and proof of 3-SAT

Mikkel

- Introduction
- Motivation
- Approximation ratio

$$\max\left(\frac{C}{C^*}, \frac{C^*}{C}\right) \leq \rho(n)$$

- Examples
  - `VERTEX-COVER`
  - `3-CNF-SAT`

Fie

- Introduction
  - What are and why do we use approximation algorithms?
- Approximation Ratio
  - What is Approximation Ratio?

- Vertex Cover
  - What is Vertex Cover?
  - Proof with example (if time)
- 3-SAT
  - What is 3-SAT?
  - Proof with example (if time)