Algorithms and Probability

Exercise Session 3

https://n.ethz.ch/~ahmala/anw

252-0030-00L Algorithmen und Wahrscheinlichkeit FS2024 / Peer Grading / Peer Grading 1 - Submission

Peer Grading 1 - Submission

Workshop | Settings | Assessment form | Submissions allocation | More

Submissions open: Thursday, 7 March 2024, 6:00 PM
Submissions close: Thursday, 14 March 2024, 10:00 AM
Assessments open: Thursday, 14 March 2024, 6:00 PM
Assessments close: Sunday, 17 March 2024, 11:59 PM

Theory Exercise From Last Week

Aufgabe 1 – *Zusammenhang*

Im Folgenden sei $G = (V, E)$ ein *zusammenhängender* Graph mit mindestens drei Knoten, d.h. $|V| \geq 3$.

(a) Beweisen Sie folgende Aussage: Wenn $deg(v)$ für alle $v \in V$ eine gerade Zahl ist, dann ist G 2-Kanten-zusammenhängend. Gilt die umgekehrte Implikation ebenfalls? Genauer: Gilt für jeden 2-Kanten-zusammenhängenden Graph $G = (V, E)$, dass für alle $v \in V$ der Grad $deg(v)$ eine gerade Zahl ist?

(b) Beweisen Sie oder widerlegen Sie die folgenden Behauptungen:

(i) Falls G einen Hamiltonkreis enthält, so ist G 2-zusammenhängend.

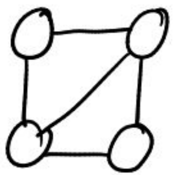
(ii) Falls G 2-zusammenhängend ist, so enthält G einen Hamiltonkreis.

(c) Nehmen Sie an, dass G 2-zusammenhängend ist. Sei (u, v, w) ein Pfad der Länge 2 in G. Zeigen Sie, dass wir diesen Pfad zu einem Kreis erweitern können, also, dass G einen Kreis enthält, in dem u, v, w als benachbarte Knoten vorkommen.

Not Worth 2 Points

a.1) Jede Grad ist gerade und G ist zusammenhängend ⇒ G enthält einen Eulerzyklus ⇒ Es gibt für jedes u-v paar 2 kantendisjunkte Pfade im Eulerzyklus ⇒ G ist 2-Kanten-zusammenhängend

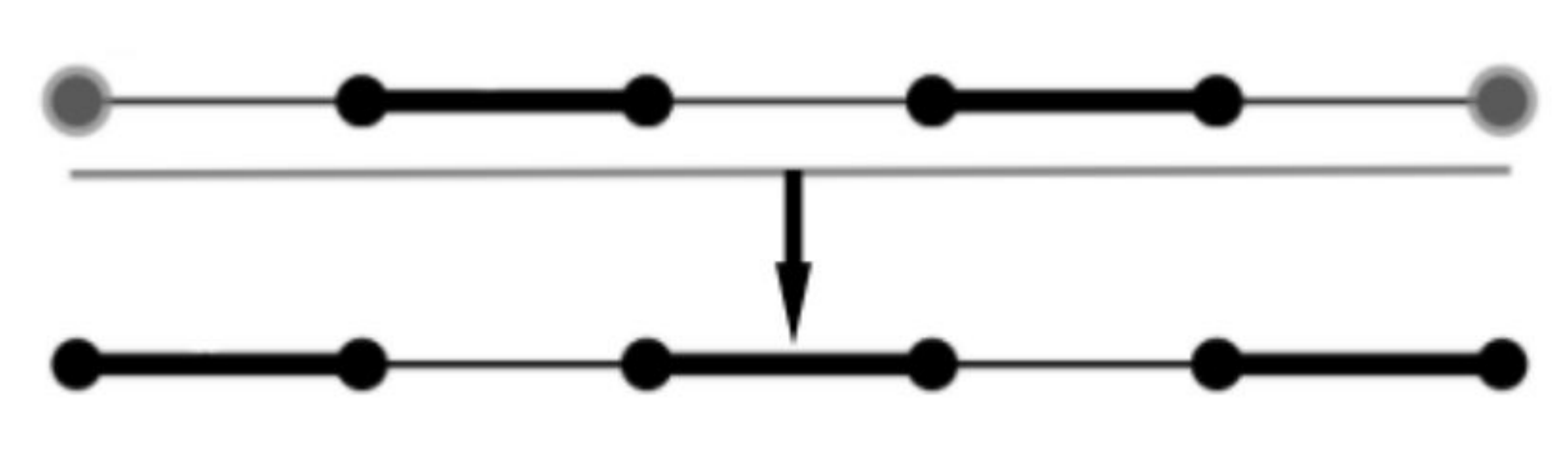
a.2) Nein, siehe:



Augmenting Path for Matching M

every second edge in the path is in M

Starting and ending vertices are not covered



Berge

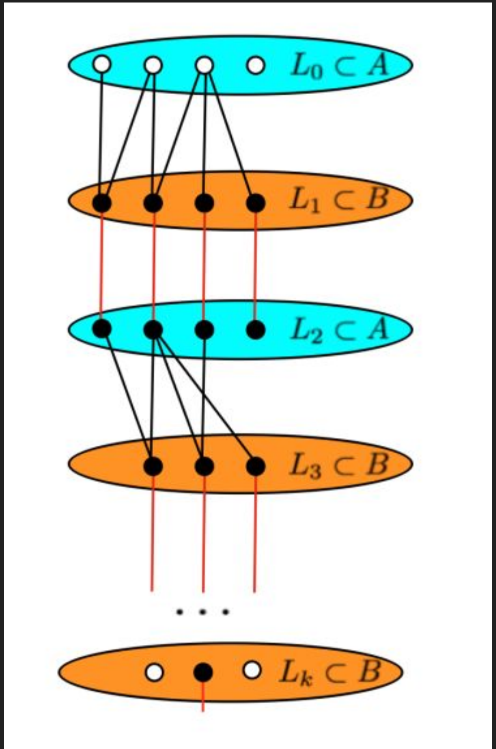
Let M be a matching in the graph $G = (V, E)$.

M is a maximum matching in G if and only if there exists no M-augmenting path in G.

Augmenting Paths in Bipartite Graphs in O(VE)

AUGMENTING_PATH $(G = (A \uplus B, E), M)$

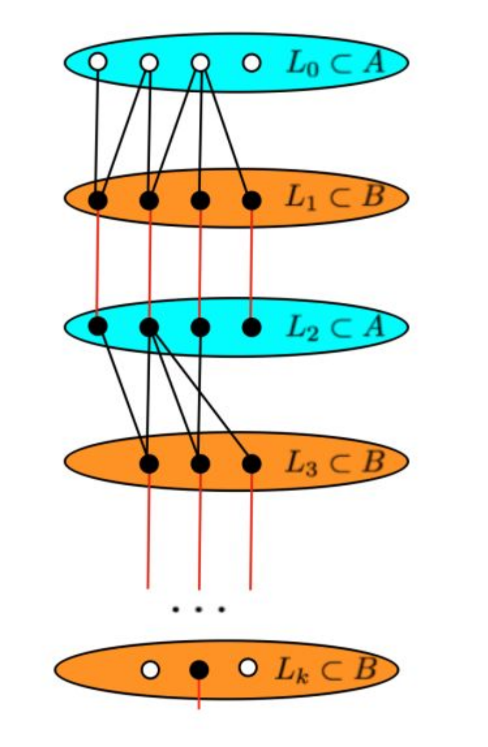
$L_0 := \{$unüberdeckte Knoten in $A\}$
Markiere alle Knoten aus L_0 als besucht.
if $L_0 = \emptyset$ **then**
    return M ist maximal
for all $i = 1$ **to** n **do**
    if i ungerade **then**
        $L_i := \{$unbesuchte Nachbarn von L_{i-1} via Kanten in $E \setminus M\}$
    else
        $L_i := \{$unbesuchte Nachbarn von L_{i-1} via Kanten in $M\}$
    Markiere alle Knoten aus L_i als besucht.
    if L_i enthält unüberdeckten Knoten v **then**
        Finde Pfad P von L_0 nach v durch backtracking
        return P // *terminiert Algorithmus*
return M ist schon maximal



Hopcroft-Karp Algorithm in $O(E\sqrt{V})$

Build an alternating level graph rooted at unmatched vertices in L using BFS.

Augment M with a maximal set of vertex-disjoint shortest-length paths using DFS.



Hall's Marriage Theorem

Let $G = (A \cup B, E)$ be a bipartite graph and let $N(X)$ denote the neighbourhood of $X \subseteq A \cup B$ in G (all vertices in $A \cup B$ adjacent to at least one vertex in X). There exists a matching that covers all vertices in A if and only if $\forall W \subseteq A : |W| \leq |N(W)|$

Satz 1.53. Sei $G = (A \uplus B, E)$ ein k-regulärer bipartiter Graph. Dann gibt es $M_1, \ldots, M_k$ so dass $E = M_1 \uplus \ldots \uplus M_k$ und alle M_i, $1 \leq i \leq k$, perfekte Matchings in G sind.

Aufgabe 1 – *Lateinisches Rechteck*

Ein lateinisches $r \times n$-Rechteck ($r \leq n$) ist eine Anordnung der Zahlen $1, \dots, n$ in r Zeilen und n Spalten, so dass in jeder Zeile und jeder Spalte jede Zahl höchstens einmal vorkommt. Ein lateinisches n-Quadrat ist ein lateinisches $n \times n$-Rechteck.

$$\begin{bmatrix} 4 & 1 & 2 & 3 \\ 2 & 3 & 1 & 4 \\ 3 & 2 & 4 & 1 \end{bmatrix}$$

Beispiel eines lateinischen 3×4-Rechtecks.

Angenommen wir haben ein lateinisches $r \times n$-Rechteck mit $r < n$ gegeben. Wir wollen sehen, ob wir es zu einem lateinischen n-Quadrat erweitern können. Das heisst, wir wollen $n - r$ weitere Zeilen mit Zahlen aus $1, \dots, n$ zu dem Rechteck hinzufügen, ohne eine Spalte oder eine Zeile in der eine Zahl mehrfach vorkommt zu erhalten.

(a) Angenommen wir haben ein lateinisches $r \times n$-Rechteck wie oben beschrieben. Wir wollen zeigen, wann man dieses zu einem $(r+1) \times n$-Rechteck erweitern kann. Beschreiben Sie, wie man dieses Problem mit einem bipartiten Graphen $G = (A \uplus B, E)$ modellieren kann und zeigen Sie, dass die Erweiterung genau dann möglich ist, wenn G ein perfektes Matching hat.

(b) Zeigen Sie, dass der in (a) konstruierte Graph regulär ist. Das heisst, zeigen Sie, dass es eine ganze Zahl k gibt, sodass alle Knoten (sowohl die Knoten in A als auch die Knoten in B) Grad genau k haben.

(c) Benutzen Sie ihr Ergebnis aus (b) um zu beschreiben, welche $r \times n$-Rechtecke man zu einem lateinischen n-Quadrat erweitern kann.

(d) Geben Sie einen Algorithmus an, der als Eingabe ein Lateinisches $r \times n$-Rechteck nimmt und es zu einem lateinischen n-Quadrat erweitert (falls ein solches existiert) und ansonsten “Nicht möglich” ausgibt.

Hinweis: In (a) kann es hilfreich sein, die Knoten in A mit ‘Spalte 1’, ‘Spalte 2’,... zu labeln und die Knoten aus B mit ‘Nummer 1’, ‘Nummer 2’,.... Was sind dann die Kanten? Was ist die entsprechende Bedeutung eines perfekten Matchings in Bezug auf das lateinische Quadrat?

Hinweis: Für (d) können Sie die Ergebnisse der vorherigen Teilaufgaben verwenden. Beachten Sie aber, dass zu einem Algorithmus auch immer eine Laufzeitanalyse und ein Korrektheitsbeweis gehören – auch wenn dies nur ein kurzer Hinweis auf eine der vorherigen Teilaufgaben ist!

https://n.ethz.ch/~ahmala/anw

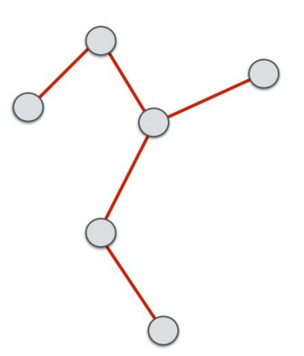
2-Approximation Algorithm for the Metric TSP

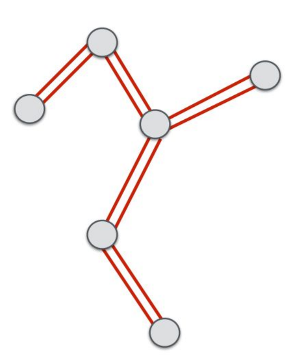
1. Compute an MST in the graph in O(n^2).

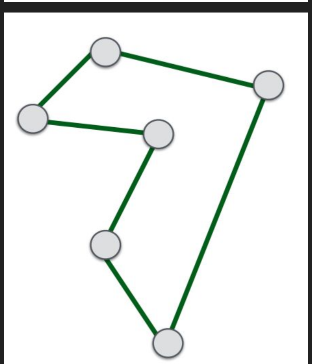
2. Create a multi-graph by doubling every edge in the MST.

3. Find an Eulerian Tour in that multi-graph

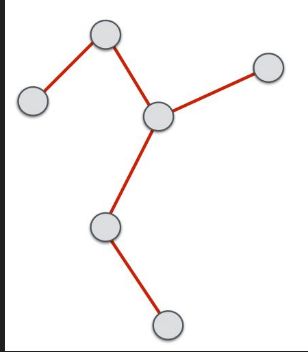
4. Translate this tour in the multi-graph to a cycle in the original graph: Use the same sequence of vertices but whenever a vertex is visited for the second time, instead go directly to the next unvisited vertex in the tour.



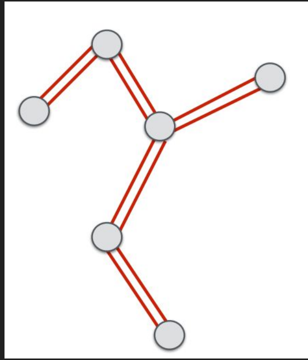




2-Approximation Algorithm for the Metric TSP

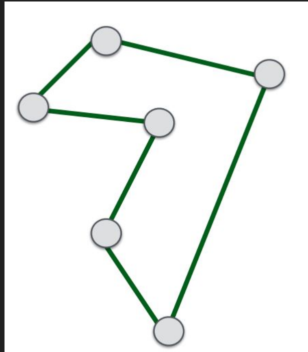


1. Compute an MST in the graph in O(n^2). $\ell(T) \leq \mathrm{opt}(K_n, \ell)$



2. Create a multi-graph by doubling every edge in the MST. $2\ell(T) \leq 2\mathrm{opt}(K_n, \ell)$

3. Find an Eulerian Tour W in that multi-graph $\ell(W) = 2\ell(T) \leq 2\mathrm{opt}(K_n, \ell)$

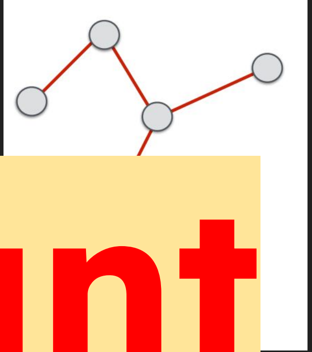


4. Translate this tour in the multi-graph to a cycle C in the original graph: Use the same sequence of vertices but whenever a vertex is visited for the second time, instead go directly to the next unvisited vertex in the tour.

$\ell(C) \leq \ell(W) = 2\ell(T) \leq 2\mathrm{opt}(K_n, \ell)$

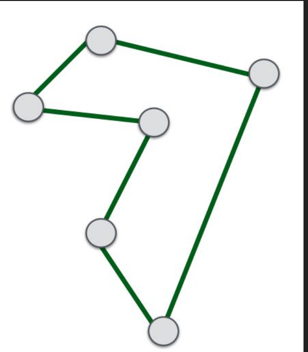
2-Approximation Algorithm for the Metric TSP

What if we want 1.5-Approximation?



4. Translate this tour in the mu … C in the original graph: Use the sam … tices but whenever a vertex is visite … me, instead go directly to the next … the tour.

$$\ell(C) \leq \ell(W) = 2\ell(T) \leq 2\mathrm{opt}(K_n, \ell)$$



2-Approximation Algorithm for the Metric TSP

2. We create a multi-graph by doubling every edge in the MST.

1.5-Approximation Algorithm for the Metric TSP

~~2. We create a multi-graph by doubling every edge in the MST.~~

2. Find a perfect matching with minimal cost in the subgraph containing only the vertices with odd degree. Then add these edges to the MST to obtain a multi-graph where all vertices have even degrees.

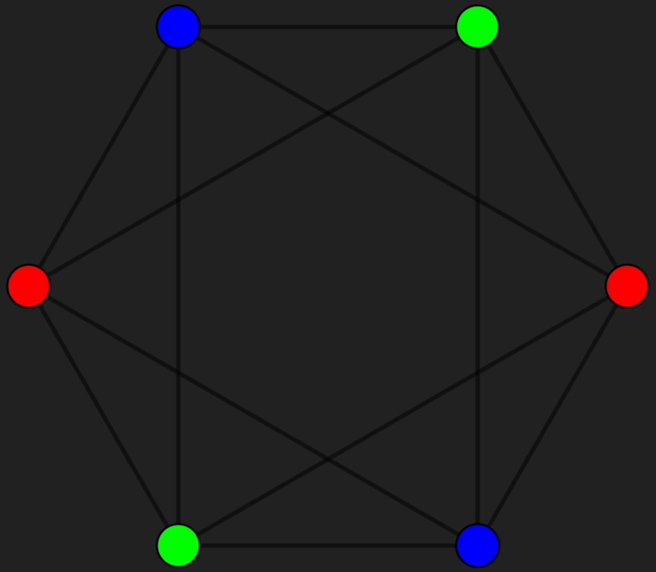
1.5-Approximation Algorithm for the Metric TSP

Satz 1.51. Für das METRISCHE TRAVELLING SALESMAN PROBLEM gibt es einen 3/2-Approximationsalgorithmus mit Laufzeit $O(n^3)$.

Coloring

In general, we define a (vertex) coloring (Färbung) of a graph $G = (V, E)$ with k colors as a mapping $c : V \to [k]$ such that $c(u) \neq c(v)$ for all edges $\{u, v\} \in E$. Moreover, we define the chromatic number (chromatische Zahl) $\chi(G)$ as the minimum number of colors needed to color G.

A graph $G = (V, E)$ is k-partite $\iff$ it can be colored with no more than k colors (i. e. $\chi(G) \leq k$).



3-partite

Kahoot

Competitive Programming Meetups

https://vis.ethz.ch/en/events/691/