SHANGHAI JIAO TONG UNIVERSITY

Algorithm Analysis HW:5

Author: *519030910100 Huangji Wang F1903004*

Course: *Fall 2021, AI2615: Algorithm*
Date: *December 24, 2021*

Problem 1 (0 points)

Given an $n \times n$ matrix A whose entries are either 0 or 1, you are allowed to choose a set of entries with value 1 and modify their value to 0. Design an efficient algorithm to decide if we can make A invertible (A is invertible if the determinant of A is not 0) by the above-mentioned modification. Prove the correctness of your algorithm and analyze its time complexity. Notice that your algorithm only needs to output yes or no and does not need to find the minimum number of modified entries. (Hint: Prove that this is possible if and only if there exist n entries with value 1 such that no two entries are in the same row and no two entries are in the same column.)

Problem 2 (25 points)

Consider the maximum flow problem $(G = (V, E), s, t, c)$ on graphs where the capacities for all edges are $1 : c(e) = 1$ for each $e \in E$. You can assume there is no pair of anti-parallel edges: for each pair of vertices $u, v \in V$, we cannot have both $(u, v) \in E$ and $(v, u) \in E$. You can also assume every vertex is reachable from s:

a. (13 points) Prove that Dinics algorithm runs in O($|E|^{3/2}$) time.

b. (12 points) Prove that Dinics algorithm runs in O($|V|^{2/3} \cdot |E|$) time. (Hint: Let f be the flow after $2|V|^{2/3}$ iterations of the algorithm. Let Di be the set of vertices at distance i from s in the residual network G^f . Prove that there exists i such that $|D_i \cup D_{i+1}| \leq |V|^{1/3}$.)

Attention: All the edges have capacity $c = 1$. No anti-parallel edges. No two-side edges. The Dinics algorithm goes like:

Algorithm 1 Dicnic Algorithm

Require: A network $G = ((V, E), c, s, t)$.

Ensure: An $s - t$ flow f of maximum value.

```
function DINICS ALGORITHM(Network G)
    ∀e ∈ E, set f(e) ← 0.
    Construct level graph G_L from G_f of G and get dist[t] from s.
    while dist[t] ≠ ∞ do
        Find a blocking flow f' ∈ G_L.
        Add augment flow f by f' and update the graph G_f.
        Construct level graph G_L from G_f of G and get dist[t] from s.
    end while
    return The total flow f.
end function
```

a. Suppose we have run Dinic algorithm after k iterations and we have already obtained a flow f. Suppose f^G is an optimal flow in the G_f. Thus, $f + f^G$ is the maximal flow. We can make sure that any path from s to t has the length k or more. We have edges in G^f can be in original graph or residual graph and thus the total capacity of G^f is at most $|E|$ and thus the flow in the residual graph should satisfy $f^G \leq \frac{|E|}{k}$. Considering the simple situation in this problem, and we can get that the number of iterations is at most $k + \frac{|E|}{k} \geq 2\sqrt{|E|}$. The equality holds if $k = \sqrt{|E|}$ and the time complexity is $O(\sqrt{|E|})$

   Secondly, the algorithm runs in the blocking flow. Based on the algorithm provided in the lecture, we can easily get that the algorithm to find the blocking flow costs $O(|E|)$. Combining these two condition and at last the total time complexity is $O(|E|^{\frac{3}{2}})$

b. Similarly, the DFS process still exists and costs $O(|E|)$. We should only consider the iteration process. Based on the hint idea we can suppose the same way in (a) and run

Dinic algorithm for k iterations and obtained a flow f. The level graph of G_f divides all vertices into different sets D_i. The current node t must be at a level at least k, meaning we have at least non-empty levels.

Now, considering we choose level L randomly from 1 to k, that is D_L. Since there are at most $|V|-1$ vertices in total across these levels, so $E[D_L] \leq \frac{|V|-1}{k}$, and we have the possibility of D_R larger than $2|V|/k$ should be less than $(|V|-1)/2|V| < 1/2$. Thus strictly more than $k/2$ levels have $D_i < 2|V|/k$ and there must be two adjacent levels j and $j+1$ for the upper bound. And we can make sure that for the union part $D_j D_{j+1} \leq 4|V|^2/k^2$ between these levels. And it can also have $|D_i \cup D_{i+1}| \leq |V|^{1/3}$. By the min-cost-max-flow theorem we make sure this is an upper bound on the flow in G_f. Thus the number of iterations is limited to $k + 4|V|^2/k^2 = \frac{1}{2}k + \frac{1}{2}k + 4|V|^2/k^2$ and get $k = 2|V|^{2/3}$ by the mean value inequality. Total time complexity is $O(|V|^{\frac{2}{3}})$. Combine each above we have the total time complexity is $O(|V|^{2/3} \cdot |E|)$.

Problem 3 (20 points)

Given an undirected and unweighted graph $G = (V, E)$, a vertex cover is a subset S of vertices such that it contains at least one endpoint of every edge, and an independent set is a subset T of vertices such that $(u, v) \notin E$ for any $u, v \in T$. Notice that V is a vertex cover, and $\emptyset$, or any set containing a single vertex, is an independent set. A minimum vertex cover is a vertex cover containing a minimum number of vertices, and a maximum independent set is an independent set containing a maximum number of vertices

a. (5 points) Prove that S is a vertex cover if and only if $V \backslash S$ is an independent set.

b. (5 points) Prove that S is a minimum vertex cover if and only if $V \backslash S$ is an maximum independent set.

c. (10 points) Design an efficient algorithm to find a minimum vertex cover, or a maximum independent set, on **bipartite graphs**. Prove the correctness of your algorithm and analyze its time complexity.

Proof:

a. **Sufficiency proof:**

By definition, we have that all the elements in S satisfy that $\forall (u, v) \in E, u \in S$ or $v \in S$.

Therefore, the set of $V \backslash S$ has the property that $\forall u, v \in V \backslash S, (u, v) \notin S$. This exactly means that we cannot find edges in the set $V \backslash S$, otherwise it will contradict if there is one edge, then at least one vertex should be in the set S. By the definition of independent set, we can conclude that $V \backslash S$ is an independent set.

Necessary proof:

By definition, the set of $V \backslash S$ has the property that $\forall u, v \in V \backslash S, (u, v) \notin E$. Therefore, $\forall (u, v) \in E, u$ **or** $v \notin V \backslash S$. We notice that the set of S is exactly a vertex cover based on the definition that at least one vertex in each edge in the set, which implies that S is a vertex cover.

Above all, S is a vertex cover if and only if $V \backslash S$ is an independent set.

b. **Proof by definition:**

S is a minimum vertex cover means that S can contain the least number of vertices in the graph $G = (V, E)$. We know that S is a vertices subset. Thus, we can have $|S| + |V \backslash S| = |S| + |V| - |S| = |V|$. So, when S is a minimum vertex cover, it means $|S|$ is the minimum number in all situations, leading to the maximum number of $|V| - |S| = |V \backslash S|$, which means $V \backslash S$ is the maximum independent set and vice versa.

c. Based on the **bipartite graphs** property, we can divide the graph into two sets:A and B.

The sets have the property that $\forall (u, v) \in E, u \in A, v \in B$ or $u \in B, v \in A$. Actually, this algorithm is finished based on the core idea of the **augmenting path**. By the definition of **bipartite graphs** we have that the maximum independent set is exactly the set of A or B, depending just on the size of them because any other vertices except of A or B

will contain at least one point in the opposite set, leading to one edge and yielding a contradiction. Therefore, we just need to find the maximum number of matching and it leads to a maximum independent set. My algorithm structure steps are:

- Divide the graph into two sets A and B.($|A| > |B|$)
- Repeat to find the correct match from A to B.
  For the current vertex $u \in A$, find $(u, v) \in E, v \in B$.
  If v isn't allocated to other vertices in A, allocating it to u.
  Else, repeating the **augmenting path** to update the allocation.
- Return the number of matching $(|A| + |B|) - N$

Time complexity: For each vertex in A, we do BFS to find the **augmenting path** in $O(|E|)$. Total number of iterations is $O(|V|)$, leading to $O(|V| \cdot |E|)$ in total.

Correctness proof:

Actually, the proof is similar to the proof of **Max-Flow-Min-Cut Theorem**. We just need to prove that the chosen vertex set can cover all the edges. We pay attention to the left and right end of each edge. All the edges we choose can have just two situations:

1. We choose the left end of the edge and the right end isn't chosen.
   Then, any edge can be covered by the left node and we don't need to choose the right node. Otherwise, it will make an edge from left node to right node.
2. We choose the right end that isn't chosen in the edge set E except the matching.
   Then, these edges must be covered by the right unrecorded vertex because it has no matching for the left end.

Actually, we don't need to consider the situation that the left end isn't chosen while the right end is chosen. In this situation, this edge is in the maximal matching. We can make sure that the right end vertex cannot be the beginning of a path by the definition of our algorithm. Thus, the left end vertex must be recorded and it is exactly the first situation. Above all, we can find the minimum vertex cover.

The algorithm goes like:

Algorithm 2 Find the maximum independent set

Require: Graph $G = (V, E)$.

Ensure: The maximum independent set.

function MATCHING($i - th$ node)
    for j from 1 to $|B|$ do **do**
        if $(i, j) \in E$ and j is not visited **then**
            j is visited
            if j has no matching or j's **ex-maching** has other augmenting path **then**
                j's new matching is i.
                return True
            end if
        end if
    end for
    return False
end function
function MAIN($G = (V, E)$)
    Run BFS on the graph to generate bipartite graph with set A and B.
    BFS : Throw odd nodes in set B and even nodes in set A.
    if B is larger than A **then**
        Swap sets A and B
    end if
    $cnt \leftarrow 0$
    for i from 1 to $|A|$ do **do**
        Initialize all the vertices not visited in B.
        $cnt \leftarrow cnt + MATCHING(i)$
    end for
    return $|A| + |B| - cnt$, $V \backslash Match_{\frac{1}{2}}$
end function

Problem 4 (20 points)

(20 points) Covert the following linear program to its standard form, and compute its dual program

$$\begin{aligned} \textbf{minimize} \quad & 2x_1 + 7x_2 + x_3 \\ \textbf{subjectto} \quad & x_1 - x_3 = 7 \\ & 3x_1 + x_2 \geq 24 \\ & x_2 \geq 0 \\ & x_3 \leq 0 \end{aligned}$$

Change **minimize** into **maximize**: $-2x_1 - 7x_2 - x_3$:

$$\begin{aligned} \textbf{maximize} \quad & -2x_1 - 7x_2 - x_3 \\ \textbf{subjectto} \quad & x_1 - x_3 = 7 \\ & 3x_1 + x_2 \geq 24 \\ & x_2 \geq 0 \\ & x_3 \leq 0 \end{aligned}$$

Change $x_3 \leq 0 \rightarrow x_4 = -x_3 \geq 0 \rightarrow x_4 \geq 0$. Reformulate:

$$\begin{aligned} \textbf{maximize} \quad & -2x_1 - 7x_2 + x_3 \\ \textbf{subjectto} \quad & x_1 + x_3 = 7 \\ & 3x_1 + x_2 \geq 24 \\ & x_2 \geq 0 \\ & x_3 \geq 0 \end{aligned}$$

Change $x_1 = x_1^+ - x_1^-, x_1^+, x_1^- \geq 0, x_1 - x_3 = x_1^+ - x_1^- - x_3 = 7$. Reformulate:

$$\begin{aligned} \textbf{maximize} \quad & -2x_1 + 2x_2 - 7x_3 + x_4 \\ \textbf{subjectto} \quad & x_1 - x_2 + x_4 = 7 \\ & 3(x_1 - x_2) + x_3 \geq 24 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

Change the equality constraint into inequality constraints, i.e.:
$x_1 - x_2 + x_4 = 7 \rightarrow x_1 - x_2 + x_4 \leq 7 \cup -x_1 + x_2 - x_4 \leq -7$

$$\begin{aligned} \textbf{maximize} \quad & -2x_1 + 2x_2 - 7x_3 + x_4 \\ \textbf{subjectto} \quad & x_1 - x_2 + x_4 \leq 7 \\ & -x_1 + x_2 - x_4 \leq -7 \\ & 3(x_1 - x_2) + x_3 \geq 24 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

Change $3(x_1 - x_2) + x_3 \geq 24$ into $-3(x_1 - x_2) - x_3 \leq -24$

$$\begin{aligned}
\textbf{maximize} \quad & -2x_1 + 2x_2 - 7x_3 + x_4 \\
\textbf{subjectto} \quad & x_1 - x_2 + x_4 \leq 7 \\
& -x_1 + x_2 - x_4 \leq -7 \\
& -3x_1 - 3x_2 - x_3 \leq -24 \\
& x_1, x_2, x_3, x_4 \geq 0
\end{aligned}$$

Actually, this is the final standard form of the LP problem.

Dual program should satisfy that:

$(\mu_1 - \mu_2 - 3\mu_3)(x_1 - x_2) - \mu_3 x_3 + (\mu_1 - \mu_2)x_4 \leq 7(\mu_1 - \mu_2) - 24\mu_3$

$(\mu_1 - \mu_2 - 3\mu_3)(x_1 - x_2) - \mu_3 x_3 + (\mu_1 - \mu_2)x_4 \geq -2x_1 + 2x_2 - 7x_3 + x_4$

$(\mu_1 - \mu_2 - 3\mu_3) \geq -2, -(\mu_1 - \mu_2 - 3\mu_3) \geq 2, -\mu_3 \geq -7, (\mu_1 - \mu_2) \geq 1$

And thus the dual problem is:

$$\begin{aligned}
\textbf{minimize} \quad & 7\mu_1 - 7\mu_2 - 24\mu_3 \\
\textbf{subject to} \quad & \mu_1 - \mu_2 - 3\mu_3 \geq -2 \\
& -\mu_1 + \mu_2 + 3\mu_3 \geq 2 \\
& -\mu_3 \geq -7 \\
& \mu_1 - \mu_2 \geq 1 \\
& \mu_1, \mu_2, \mu_3 \geq 0
\end{aligned}$$

Problem 5 (35 points)

(35 points) In this question, we will prove König-Egerváry Theorem, which states that, in any bipartite graph, the size of the maximum matching equals to the size of the minimum vertex cover. Let $G = (V, E)$ be a bipartite graph.

(a) (5 points) Explain that the following is an LP-relaxation for the maximum matching problem.

$$\begin{aligned} \text{maximize} \quad & \sum_{e \in E} x_e \\ \text{subject to} \quad & \sum_{e:e=(u,v)} x_e \leq 1 && (\forall v \in V) \\ & x_e \geq 0 && (\forall e \in E) \end{aligned}$$

(b) (5 points) Write down the dual of the above linear program, and justify that the dual program is an LP-relaxation to the minimum vertex cover problem.

(c) (10 points) Show by induction that the *incident matrix* of a bipartite graph is totally unimodular. (Given an undirected graph $G = (V, E)$, the incident matrix A is a $|V| \times |E|$ zero-one matrix where $a_{ij} = 1$ if and only if the i-th vertex and the j-th edge are incident.)

(d) (10 points) Use results in (a), (b) and (c) to prove König-Egerváry Theorem.

(e) (5 points) Give a counterexample to show that the claim in König-Egerváry Theorem fails if the graph is not bipartite.

(a) We can define that $x_e, e = (u, v)$ has the value of 0 or 1, which means that whether the vertex u matches the vertex v. By the matching problem ,it must have the restriction that at most one edge can be chosen in all the edges starting at u, which implies that $\Sigma_{e:e=(u,v)} x_e \leq 1$ Also, the edge can be chosen or not. Thus, $x_e \geq 0$. Finally, we want to have maximum matching, and that means we want to maximize the choosing method x_e based on all the edges, i.e. $\max_e x_e$.

(b) The dual form goes like:

$$\begin{aligned} \textbf{minimize} \quad & \Sigma_i a_i + \Sigma_j b_j \\ \textbf{subject to} \quad & \forall e : e = (i, j), a_i + b_j \geq 1 \\ & a_i \geq 0, b_j \geq 0 \end{aligned}$$

We define a_i is the selection situation of i-th node in the first set and b_i means the second set. We know that the minimum vertex cover problem has some properties.

First, for each edge we should make sure it can contain at least one endpoint, and this means that $\forall e : e = (i, j), a_i + b_j \geq 1$. Each node can be chosen once or zero time(not chosen) so the restriction is $w_v \geq 0$. At last, we want the selection vertices to be as small as possible, i.e., $\min \Sigma_i a_i + \Sigma_j b_j$. Above all, this is exactly an LP-relaxation to the minimum vertex cover problem.

(c) We use the induction method to prove that **the incident matrix of a bipartite graph is totally uni-modular**, i.e., we need to prove that all the $r \times r$ sub-matrices of the incident matrix will have determinant 1, 0 or 1.

1. Base step: $k = 1$.

   There is just one element in the incident matrix and it has exactly two values to be chosen. The first is 0 while the second is 1. Therefore, the determinant is 0,1 and it satisfies the problem.

2. Assumption step: $k = n \in N, n \geq 1$.

   We assume that the incident matrix of a bipartite graph is k-uni-modular, i.e. $\forall k \in [1, n]$, all the $k \times k$ sub-matrices have the determinant -1, 0 or 1.

3. Induction step: $k = n + 1 \in N$.

   We just need to consider three situations when the matrix is enlarged by a bound:

   i) The added part has one column of only zeros.
   By the definition of determinant, we can choose the column and therefore the result can lead to 0, which satisfies the assumption.

   ii) The new sub-matrix has one column with single 1.
   Suppose the place is $a(i, j) = 1$. We can choose the **1** item, take away the i-th column and j-th row and compute the sub-sub-matrix. the sub-sub-matrix is $n \times n$ and by the assumption we can make sure the result is also in -1, 0, 1, which satisfies the assumption.

   iii) All columns of the new sub-matrix have two 1 elements.
   We don't need to consider more **1 elements** situation by definition of incident matrix. We can denote S is the new sub-matrix and v as the vector that includes item 1 when the i-th row can connect to a new vertex in the former part of the new bipartite. The item -1 means by contrast. vS can be used to define a vector whose components are $\forall e : (i, j), v_i + v_j$ and its column ends up in S. However, by the definition of bipartite, for each active edge $e = (i, j)$, one is only in the former part and the other is in the later part, which means $v_i + v_j = 0$(the 1 for former and -1 for later). Thus, by the definition, vS is linear-independent and the determinant can be 0, which satisfies our assumption.

4. Above all, our proof is done.

(d) The main proof method is based on the strong duality and totally unimodular bipartite

graph. We need to prove that for the bipartite $G = (V, E)$, the number of the maximum matching edges will be equal to the minimum vertex cover.

Firstly, based on the totally uni-modular incident matrix, we can make sure that to maximize matching and to minimize vertex cover are actually dual equal to each other based on problem (a) and (b). Therefore, we just need to consider the relaxation LP problems. Besides, we consider that the truth that the primal constraint matrix is the transpose of the dual constraint matrix based on Microsoft paper about LP dual proof, the totally uni-modular incident matrix stays no change.

Next, to apply the strong duality property, we should make sure that the optimal integer can only be 0 or 1. If one of the integer solution is over 1, then it cannot be an optimal solution. The reason is that for any condition restrictions with $a_i + b_j \geq 1$, if the solution for a_i or b_j is larger than 1, then it can always reduce to 1 and it will not violate the restrictions, leading to the smaller result. Thus, it's obvious that the solution can only be 0 or 1. Then, we can apply the property of strong duality. By the property of strong duality, the final solution is equal in the primal and dual problems and thus the theorem is true.

(e) Actually, we can just consider the triangle that is not bipartite. Suppose the graph is $G = (V, E), V = \{a, b, c\} and E = \{(a, b), (b, c), (c, a)\}$. The maximum matching is obviously 1 while the minimum vertex cover contains 2 vertices. Thus, this theorem cannot be well applied in the non-bipartite graphs.

Problem 6

How long does it take you to finish the assignment (include thinking and discussing)? Give a score (1,2,3,4,5) to the difficulty. Do you have any collaborators? Write down their names here.

I take 14 hours in total to finish the assignment(include thinking and discussing).

Difficulty: 5.

Collaborators: Xiangge Huang in Problem 2, 4, 5.

I'm not familiar with the knowledge of network flow and LP problems. Also, the NP lecture let me puzzled. Even if Mr. Tao said the problem is easy, I'm afraid whether the task difficulty in the final exam will beat me down.