CMPUT 675 - Assignment #4
Saeed Najafi, 1509106

Winter 2021, University of Alberta

Pages: 9

Exercise 1 - Inflating Feasible Polytopes to Have Large(ish) Size

Marks: 2

Let $A \in \mathbb{Z}^{m \times n}, b \in \mathbb{Z}^m$ and Γ be the maximum absolute value of all entries in A and b (as in the lectures). Let $\delta := \frac{1}{2\max\{n,m\}\cdot(n\Gamma)^n}$, also as in the lectures.

Show if $\overline{x} \in \mathbb{R}^n$ satisfies $A\overline{x} \leq b$, then for any z satisfying $z_j \in [x_j - \delta^2, x_j + \delta^2]$ for each $1 \leq j \leq n$, $Az \leq b'$ where $b'_i = b_i + \delta$ for each $1 \leq i \leq m$. Conclude the volume of solutions satisfying $Ax \leq b'$ is at least $(2\delta^2)^n$.

Note: One could prove this for a slightly larger δ. Feel free to prove it for something other than δ^2 as long as the inverse logarithm of that value is bounded by a polynomial in n and $\log \Gamma$.

Note: Just verify it directly, this isn't supposed to be conceptually hard but make sure you write the details carefully and accurately.

Solution 1: We can assume that $\frac{1}{2\max\{n,m\}\cdot(n\Gamma)^n} \leq \frac{1}{n\Gamma}$. Hence, $n\Gamma\delta \leq 1$ and $n\Gamma\delta^2 \leq \delta$. Assume there is a z such that $|z_j - x_j| \leq \delta^2$ for each $1 \leq j \leq n$. Since Γ is the maximum absolute value of all entries in A, then $A \cdot (z - x) \leq n\Gamma\delta^2\mathbf{1} \leq \delta\mathbf{1}$. Therefore, if $x \in \mathbb{R}^n$ satisfies $Ax \leq b$, then z satisfies $Az = Ax + A(z - x) \leq b + \delta\mathbf{1}$. Therefore, all such z satisfies $Az \leq b'$. If such z can be picked for all dimensions of x, then it is from an n-dimensional cube with side length $2\delta^2$ around a potential x. Therefore, the convex body $Ax \leq b'$ must include such n-dimensional cube. We can conclude that the volume of $Ax \leq b'$ is at least $(2\delta^2)^n$.

Exercise 2 - Farkas Lemma with Small Support Size

Marks: 3

Let $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$.

- Show that in any extreme point $\overline{y}$ of the set $\{y \in \mathbb{R}^m : A^T \cdot y \geq \mathbf{0}, y \geq 0, b^T \cdot y = -1\}$ we have that $\overline{y}_i \neq 0$ for at most $n + 1$ indices $1 \leq i \leq m$.

  Solution 2.1: Let's say there is an extreme point which has at least $n + 2$ non-zero values in different dimensions. Then, we can have at most $m - n - 2$ zero values in the other dimensions of the $\overline{y}_i$. Considering the non-negative constraints as part of the equality constraints and the equality constraint $b^T \cdot \overline{y} = -1$, the matrix A^* in the equality constraints $A^*\overline{y} = (\frac{\mathbf{0}}{-1})$ can have at most $n+1+m-n-2$ rows: n rows from $-A^T$, 1 row for b, and at most $m-n-2$ rows from $I_{m \times m}$. Hence, A^* can have at most $m-1$ rows, but it has m columns, and its rank must be less than m (rank $\leq \min(m-1, m)$), but this contradicts the fact that $\overline{y}$ is an extreme point as the rank(A^*) must be exactly m.

- Show that if $\mathcal{P} = \emptyset$ where $\mathcal{P} = \{x \in \mathbb{R}^n : A \cdot x \leq b, x \geq 0\}$ then in fact there is a collection of $m' \leq n+1$ constraints $A' \cdot x \leq b'$ from the system $A \cdot x \leq b$ such that $\mathcal{Q} = \emptyset$ where $\mathcal{Q} = \{x \in \mathbb{R}^n : A' \cdot x \leq b', x \geq 0\}$.

  Solution 2.2: If $\mathcal{P} = \emptyset$, according to the Farkas Lemma and the previous part, there exists an extreme point $\overline{y}$ for the set $\{y \in \mathbb{R}^m : A^T \cdot y \geq \mathbf{0}, y \geq 0, b^T \cdot y = -1\}$ such that $m^{'} \leq (n+1)$ dimensions of $\overline{y}$ are non-zero. Let's name these dimensions $i_1, i_2, ..., i_{m'}$. Let's consider the row vector $\overline{y}^{'T} = (\overline{y}_{i_1}, \overline{y}_{i_2}, ..., \overline{y}_{i_{m'}})$, the row vector $b^{'T} = (b_{i_1}, b_{i_2}, ..., b_{i_{m'}})$, and the matrix $A^{'T} = (A_{i_1}|A_{i_2}|...|A_{i_{m'}})$ where A_j is the row j of the matrix A. Since the other dimensions of $\overline{y}$ are zero, then $b^{'T}\overline{y}^{'} = -1$, $\overline{y}^{'} \geq 0$, and $A^{'T}\overline{y}^{'} = A_{i_1}\overline{y}_{i_1} + A_{i_2}\overline{y}_{i_2} + ... + A_{i'_m}\overline{y}_{i_{m'}} \geq 0$. Therefore, the set $\{y^{'} \in \mathbb{R}^{m^{'}} : A^{'T} \cdot y^{'} \geq \mathbf{0}, y^{'} \geq 0, b^{'T} \cdot y^{'} = -1\}$ is not empty, and according to the Farkas Lemma, the set $\mathcal{Q} = \{x \in \mathbb{R}^n : A' \cdot x \leq b', x \geq 0\}$ is now empty.

Ultimately, this means in the step where we slightly relax the constraints by some tiny $\delta > 0$ when solving for feasibility via the Ellipsoid method, it suffices to let δ depend on the maximum bit complexity of coefficients in the constraints and n (but not on m).

Exercise 3 - A Duality Argument for Maximum Spanning Trees

Marks: 5

Let $G = (V, E)$ be an undirected graph with edge costs $c(e) \geq 0$. Consider the maximum-cost spanning tree problem where we are to find a spanning tree $T \subseteq E$ of maximum possible cost $c(T)$.

Consider the following LP relaxation where we have a variable x_e for each edge $e \in E$.

$$\begin{array}{rrcll} \textbf{maximize}: & \sum_{e \in E} c(e) \cdot x_e & & & \\ \textbf{subject to}: & x(E[S]) & \leq & |S| - 1 & \forall \emptyset \subsetneq S \subseteq V \\ & x & \geq & 0 & \end{array}$$

Observe in this case we do not have constraint $x(E) = |V| - 1$, but we do have a constraint for $S = V$ which is really saying $x(E) \leq |V| - 1$.

- Write the dual of this LP. You can ask me if you have the correct dual before submitting the whole assignment. I will grade this part before you proceed (since you must have the correct dual LP for the next part).

  Solution 3.1: For every subset of nodes S such that $\emptyset \subsetneq S \subseteq V$, we consider a dual variable Y_S. The dual problem becomes:

$$\begin{array}{rlcll} \textbf{minimize}: & \sum_S (|S| - 1) \cdot Y_S \quad \emptyset \subsetneq S \subseteq V & & & \\ \textbf{subject to}: & \sum_{S: e \in E[S]} Y_S & \geq & c(e) & \forall e \in E \\ & Y_S & \geq & 0 & \emptyset \subsetneq S \subseteq V \end{array}$$

- Now describe how to construct a feasible dual LP solution whose value equals the value of a maximum spanning tree: of course you must prove it is feasible and that its value equals the value of the maximum spanning tree.

  Solution 3.2: We use the Kruskal's algorithm to build an optimal solution for the maximum spanning tree problem. We assume that it can be proved that the algorithm correctly finds the maximum spanning tree edges given its greedy nature. Based on the iterations of the algorithm, we form a dual solution and check the complimentary slackness conditions. Therefore, the Kruskal's algorithm can build such an optimal solution for the dual problem. We assume the following iterations from the Kruskal's algorithm:

  – Pick an edge e with the maximum weight.

  – Use edge e in the tree if it connects two separate connected components in the tree without causing any cycles.

  We assume that during the algorithm, we are picking some of the original edges. Let's name these edges $e_1, e_2, ..., e_{m'}$. Let S_i be the set of nodes corresponding to the connected component built from merging e_i into the forest with edges $e_1, e_2, ..., e_{i-1}$. Therefore, throughout the algorithm, we have $m^{'}$ connected components $S_1, S_2, ..., S_{m'}$. Define $Y_{S_i} = c(e_i) - c(e_b)$ $(\forall 1 \leq i \leq m^{'} - 1)$, $Y_{S_{m'}} = c(e_{m'})$, and $Y_S = 0$ for the rest of node subsets. The index b is the smallest index greater than i such that both endpoints of e_i are in S_b. Since $b > i$, then $c(e_i) \geq c(e_b)$, hence $Y_S \geq 0$ for every node subset S. For the edges $e_1, e_2, ..., e_{m'}$, the first dual constraint is tight: if $e = e_i$, then $\sum_{S:e_i \in E[S]} Y_S = c(e_i) - c(e_b) + c(e_b) - c(e_{b'}) + ... - c(e_{m'}) + c(e_{m'}) = c(e_i)$ ($S_{m'-1}$ and $S_{m'}$ must include e_i as they become full spanning trees). If one of the original edges of the graph e_a has not been picked by the algorithm, there must be a connected component S_j (smallest j) such that both endpoints of e_a are already in S_j (i.e. adding e_a causes a cycle in S_j). Therefore, we have $\sum_{S:e_a \in E[S]} Y_S = c(e_j) \geq c(e_a)$ as the algorithm picks e_j before checking e_a $(c(e_j) \geq c(e_a))$. We can conclude that such Y is a feasible solution for the dual problem.

  To also prove that this mentioned Y is an optimal solution for the dual problem, we can check the complimentary slackness conditions. For the picked edges from the Kruskal's algorithm $e_1, e_2, ..., e_{m'}$, we have $x_e > 0$, but the dual constraint is tight. If $Y_S > 0$, then we must have an index k such that $S = S_k$ and S_k is a connected component without any cycles, therefore $x(E[S_k]) = |S_k| - 1$ and now the prime constraint is tight. If e_a has not been picked by the algorithm, then the dual constraint for e_a can be non-tight if $c(e_j) > c(e_a)$ (j from previous paragraph), however, $x_{e_a} = 0$ and the complimentary slackness condition $x_{e_a} \cdot (\sum_{S:e_a \in E[S]} Y_S - c(e_a))$ still holds.

  Hint: It will help to remind yourself how Kruskal's algorithm works for computing maximum-cost spanning trees (hint: it is the same as minimum spanning trees but you process edges in decreasing order of cost). Write the complementary slackness conditions to get a further hint on how you can proceed. Do some small examples by hand (eg. start with a triangle).

Exercise 4 - Directed Cuts

Marks: 5

Consider the following generalization of the min-cost arborescence problem. You are given a directed graph $\mathcal{G} = (V, E)$ with edge costs $c(e) \geq 0$, a root node $r \in V$, and an integer connectivity requirement $k \geq 1$. The goal is to find the minimum-cost subset $F \subseteq E$ such in the subgraph (V, F), for each $v \in V - \{r\}$, there are at least k edge-disjoint subpaths from r to v. Of course, as you know this is equivalent to $|\delta^{in}(S) \cap F| \geq k$ for every $\{v\} \subseteq S \subseteq V - \{r\}$ (by max-flow / min-cut).

Consider the following LP relaxation where we have a variable x_e for each $e \in E$.

$$\begin{array}{rrcll} \textbf{minimize}: & \sum_{e \in E} c(e) \cdot x_e & & & \\ \textbf{subject to}: & x(\delta^{in}(S)) & \geq & k & \forall\, \emptyset \subsetneq S \subseteq V - \{r\} \\ & x_e & \leq & 1 & \forall\, e \in E \\ & x & \geq & 0 & \end{array}$$

Note, this LP has a simple separation oracle: first check $x \geq 0$ and, if so, for each $v \in V - \{r\}$ simply determine if the minimum $r - v$ cut is $\geq k$ where we view $x_e, e \in E$ as giving the capacities of the edges (outputting a violated cut if the maximum $r - v$ flow is $< k$).

You will show this LP is integral (i.e. extreme points are integral). The ultimate plan is to "uncross" the constraints so that you see a TUM system of tight constraints forming a basis for the tight constraints.

Recall for a subset of edges F we let $\chi_F \in \{0,1\}^E$ denote the indicator vector for F: it takes value 1 for $e \in F$ and value 0 for $e \notin F$.

- Show for any $A, B \subseteq V$ and any $c : E \to \mathbb{R}_{\geq 0}$ that $c(\delta^{in}(A)) + c(\delta^{in}(B)) \geq c(\delta^{in}(A \cap B)) + c(\delta^{in}(A \cup B))$. That is, the *directed* cut function is submodular (we only showed it for undirected cuts when we discussed Gomory-Hu Trees). Note this is for any pair of subsets, even (perhaps) empty ones or ones that include r.

  Solution 4.1: Consider an edge $e = uv$. Then we have the following cases:

  - Case 1) $u \notin A \cup B$, but $v \in (A - B)$.

    Then $e \in \delta^{in}(A)$ and $e \in \delta^{in}(A \cup B)$, but $e \notin \delta^{in}(B)$ and $e \notin \delta^{in}(A \cap B)$. We have $c(e) \geq c(e)$.

  - Case 2) $u \notin A \cup B$, but $v \in (B - A)$.

    Similar to Case 1, the claim holds.

  - Case 3) $u \notin A \cup B$, but $v \in A \cap B$.

    Then $e \in \delta^{in}(A)$, $e \in \delta^{in}(B)$, $e \in \delta^{in}(A \cap B)$, and $e \in \delta^{in}(A \cup B)$. We have $2c(e) \geq 2c(e)$.

  - Case 4) $u \in (A - B)$ and $v \in (A - B)$.

    Then $e \notin \delta^{in}(A)$, $e \notin \delta^{in}(B)$, $e \notin \delta^{in}(A \cap B)$, and $e \notin \delta^{in}(A \cup B)$. We have $0 \geq 0$.

  - Case 5) $u \in (B - A)$ and $v \in (B - A)$.

    Similar to Case 4, the claim holds.

  - Case 6) $u \in (A - B)$ and $v \in (B - A)$.

    Then $e \notin \delta^{in}(A)$, $e \in \delta^{in}(B)$, $e \notin \delta^{in}(A \cap B)$, and $e \notin \delta^{in}(A \cup B)$. We have $c(e) \geq 0$.

- Case 7) $u \in (B - A)$ and $v \in (A - B)$.

  Similar to Case 6, the claim holds.

- Case 8) $u \in (A - B)$ and $v \in (A \cap B)$.

  Then $e \notin \delta^{in}(A)$, $e \in \delta^{in}(B)$, $e \in \delta^{in}(A \cap B)$, and $e \notin \delta^{in}(A \cup B)$. We have $c(e) \geq c(e)$.

- Case 9) $u \in (B - A)$ and $v \in (A \cap B)$.

  Similar to Case 8, the claim holds.

- Case 10) $u \notin (A \cup B)$ and $v \notin (A \cup B)$.

  Then $e \notin \delta^{in}(A)$, $e \notin \delta^{in}(B)$, $e \notin \delta^{in}(A \cap B)$, and $e \notin \delta^{in}(A \cup B)$. We have $0 \geq 0$.

- Next, let $\overline{x}$ be an extreme point of the LP above. Assume $\overline{x}_e \neq 0$ for each $e \in E$ (otherwise we could use induction to show $\overline{x}$ is integral, you can skip this small argument). You can carry this assumption through all steps below.

  Let A, B be subsets of V corresponding to tight constraints such that $A \cap B \neq \emptyset$. Show: 1) $x(\delta^{in}(A \cap B)) = x(\delta^{in}(A \cup B)) = k$ and 2) we further have $\chi_{\delta^{in}(A)} + \chi_{\delta^{in}(B)} = \chi_{\delta^{in}(A \cap B)} + \chi_{\delta^{in}(A \cup B)}$.

  Solution 4.2: Since both A, B are subsets of V corresponding to tight constraints, we have $\overline{x}(\delta^{in}(A)) = k$ and $\overline{x}(\delta^{in}(B)) = k$. Considering $\overline{x}$ as the capacity function and according to the previous section, $\overline{x}(\delta^{in}(A)) + \overline{x}(\delta^{in}(B)) \geq \overline{x}(\delta^{in}(A \cap B)) + \overline{x}(\delta^{in}(A \cup B))$. Hence, $\overline{x}(\delta^{in}(A \cap B)) + \overline{x}(\delta^{in}(A \cup B)) \leq 2k$. Since $\overline{x}$ is an extreme point and $A \cap B \neq \emptyset$ and $A \cup B \neq \emptyset$, we also have $\overline{x}(\delta^{in}(A \cap B)) \geq k$ and $\overline{x}(\delta^{in}(A \cup B)) \geq k$. Therefore, $\overline{x}(\delta^{in}(A \cap B)) + \overline{x}(\delta^{in}(A \cup B)) \geq 2k$ and we can conclude that $\overline{x}(\delta^{in}(A \cap B)) + \overline{x}(\delta^{in}(A \cup B)) = 2k$. This final equation and the fact that $\overline{x}(\delta^{in}(A \cap B)) \geq k$ and $\overline{x}(\delta^{in}(A \cup B)) \geq k$, we must have $\overline{x}(\delta^{in}(A \cap B)) = k$ and $\overline{x}(\delta^{in}(A \cup B)) = k$.

  Consider the argument presented in the **Solution 4.1**. The only cases where the LHS is greater than the RHS are the Cases 6 and 7, where there is an edge with one endpoint in $A-B$ (or $B-A$), and another endpoint in $B-A$ (or $A-B$). But these cases contradicts the fact from the previous paragraph that $\delta^{in}(A \cap B) = \delta^{in}(A \cup B) = \delta^{in}(A) = \delta^{in}(B)$. Therefore, we always have the equality for our cost function $\overline{x}$. This means that if we consider the value of the dimension of each edge in $\chi_{\delta^{in}(A)} + \chi_{\delta^{in}(B)}$, it must be the same as its value in $\chi_{\delta^{in}(A \cap B)} + \chi_{\delta^{in}(A \cup B)}$.

- **Just a Comment**: One could take the last step further to argue there is a laminar family $\mathcal{L} \subseteq 2^{V - \{r\}}$ of subsets of $V - \{r\}$ such that: 1) each $A \in \mathcal{L}$ is tight, meaning $x(\delta^{in}(A)) = k$, and 2) the characteristic vectors $\{\chi_{\delta^{in}(A)} : A \in \mathcal{L}\}$ form a basis for the set $\{\chi_{\delta^{in}(S)} : \emptyset \subsetneq S \subseteq V - \{r\}, x(\delta^{in}(S)) = k\}$ (i.e. for the tight cut constraints). But this proof would be the exact same as in the lecture for the spanning tree polytope so you don't have to do it.

- Finally, argue the matrix corresponding to the $\{\chi_{\delta^{in}(A)} : A \in \mathcal{L}\}$ is a network matrix, thus it is TUM. Conclude $\overline{x}$ is integral.

  Solution 4.3: The solution is similar to the argument presented in the lecture. Consider the tree structure corresponding to the laminar family $\mathcal{L}$. So for every edge e, define an ordered pair $P_e = (A, X)$ where A is the smallest set with $e \in \delta^{in}(A)$. (if e does not appear in any sets, then let $P_e = (X, X)$). Then the network matrix for the tree T and ordered pairs

$\{P_e : e \in E\}$ is exactly equal to the constraint matrix corresponding to the $\{\chi_{\delta^{in}(A)} : A \in \mathcal{L}\}$ (one row for each $\chi^T_{\delta^{in}(A)}$). Therefore, this constraint matrix is a network matrix and it is TUM. As the rows are also the bases for the space of the tight constraints, then the matrix is a full rank of tight constraints, and according to the Cramer's rule, every dimension of the extreme points is integral.

Naturally, there are combinatorial algorithms for this problem as well. They rely on **matroid intersection** techniques.

Exercise 5 - Tracing the Min-Cost Arborescence Algorithm

Marks: 4

Compute a minimum-cost arborescence (rooted at r) in the above directed graph. In each iteration:

- Describe the set of edges F and the cycles of F.
- Describe the new graph where each vertex is labelled by the set of original nodes that were contracted to that vertex.
- Describe the new costs c' of the remaining edges after contracting the cycles.

If there is ever a tie for the cheapest edge entering a vertex, break ties however you want.

Once you finally reach an iteration where F is acyclic, step back through the graphs that you contracted to show the arborescences you buy in each such graph.

Also give a dual solution y witnessing the fact that your arborescence is a minimum-cost arborescence. You do **not** have to track the construction of y throughout the entire algorithm (though it might be helpful). Of course, you can just give the values for y_S that are nonzero.

Solution 5: The initial tree T is empty and $Y_S = 0$ ($\forall\, \emptyset \subsetneq S \subseteq \{a, b, c, d, e, f, g, i, j\}$). We have the initial partitioning $P_0 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{g\}, \{e\}, \{f\}, \{i\}, \{j\}\}$. We will have three iterations for the first phase of the algorithm where we raise values of each subset until one edge becomes tight and merge potential cycles.

- Phase 1, iteration 1 (Figure 1):

  - Raise $Y_{\{a\}} = 2$ and Add ba to T

  - Raise $Y_{\{b\}} = 3$ and Add ab to T

  - Raise $Y_{\{c\}} = 4$ and Add bc to T

  - Raise $Y_{\{d\}} = 2$ and Add gd to T

  - Raise $Y_{\{g\}} = 3$ and Add eg to T

  - Raise $Y_{\{e\}} = 2$ and Add de to T

  - Raise $Y_{\{f\}} = 4$ and Add gf to T

  - Raise $Y_{\{i\}} = 4$ and Add gi to T

  - Raise $Y_{\{j\}} = 3$ and Add ij to T

  - $P_1 = \{\{a, b\}, \{c\}, \{d, e, g\}, \{f\}, \{i\}, \{j\}\}$

- $T = \{ba, ab, bc, gd, eg, de, gf, gi, ij\}$

- Phase 1, iteration 2 (Figure 2):

  - Raise $Y_{\{a,b\}} = 3$ and Add ra to T

  - Raise $Y_{\{d,e,g\}} = 3$ and Add fe to T

  - $P_2 = \{\{a, b\}, \{c\}, \{d, e, g, f\}, \{i\}, \{j\}\}$

  - $T = \{ba, ab, bc, gd, eg, de, gf, gi, ij, ra, fe\}$

- Phase 1, iteration 3 (last) (Figure 3):

  - Raise $Y_{\{d,g,e,f\}} = 1$ and Add cd to T

  - no extra cycles and T is feasible.

  - $P_3 = \{\{a, b\}, \{c\}, \{d, e, g, f\}, \{i\}, \{j\}\}$

  - $T = \{ba, ab, bc, gd, eg, de, gf, gi, ij, ra, fe, cd\}$

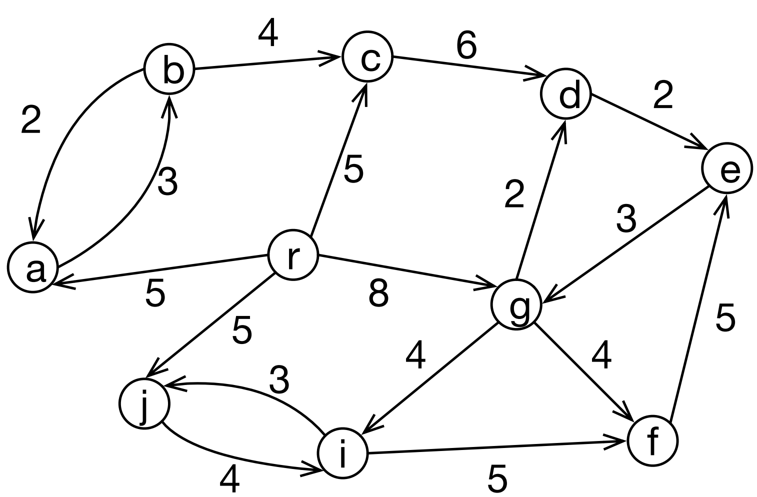
- Phase 2 (Figure 4):

  - Remove edges in the tree in the reverse order they have been added as long as the tree stays feasible.

  - $T = \{ab, bc, eg, de, gf, gi, ij, ra, cd\}$

- Optimal solution for the dual problem, Y_S is zero except these:

  - $Y_{\{a,b\}} = 3$, $Y_{\{d,e,g,f\}} = 1$, $Y_{\{d,e,g\}} = 3$, $Y_{\{j\}} = 3$, $Y_{\{f\}} = 4$, $Y_{\{i\}} = 4$, $Y_{\{e\}} = 2$, $Y_{\{g\}} = 3$, $Y_{\{c\}} = 4$, $Y_{\{d\}} = 2$, $Y_{\{a\}} = 2$, $Y_{\{b\}} = 3$.



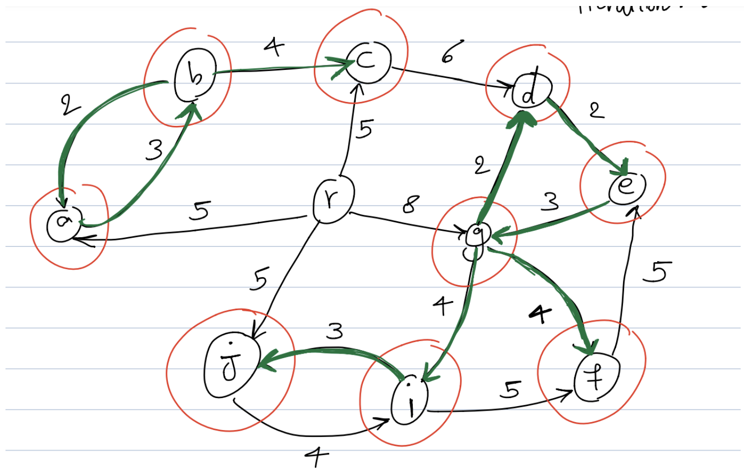


Figure 1: Iteration 1 of phase 1.

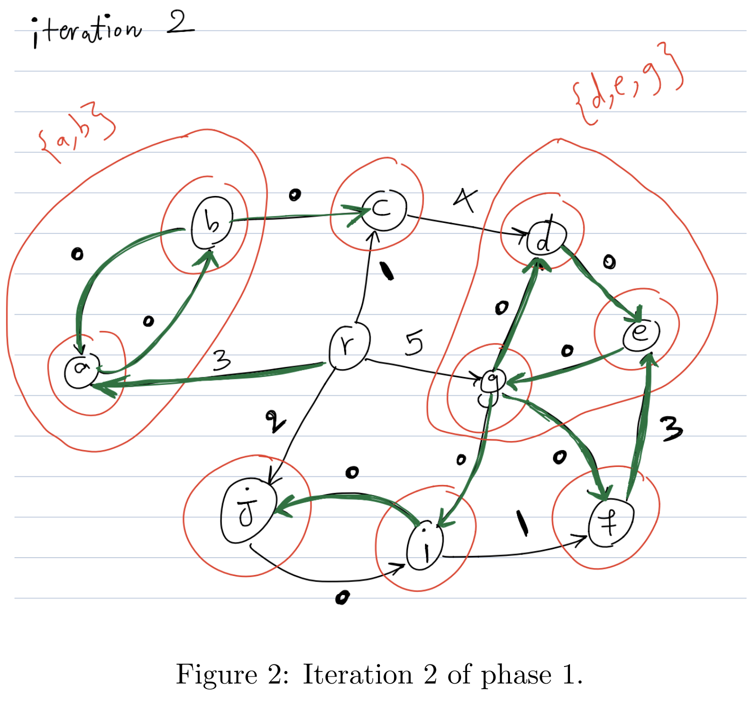


Figure 2: Iteration 2 of phase 1.

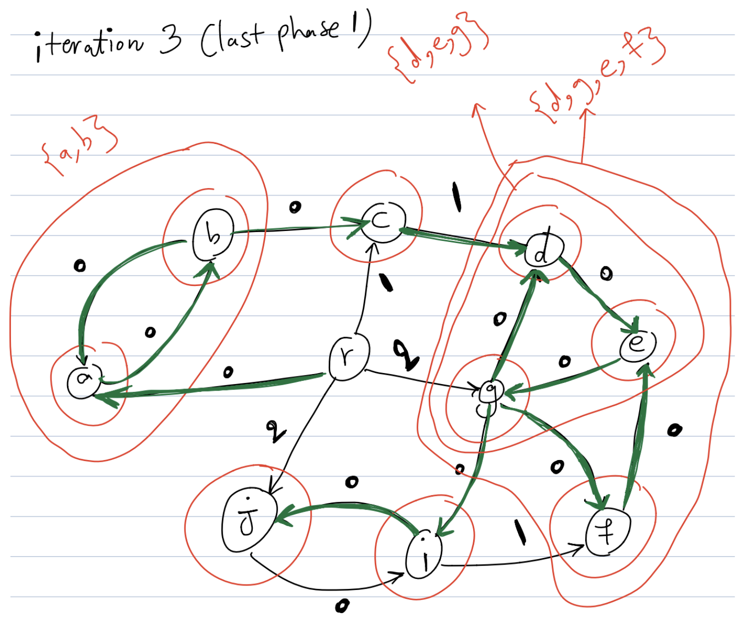


Figure 3: Iteration 3 of phase 1.

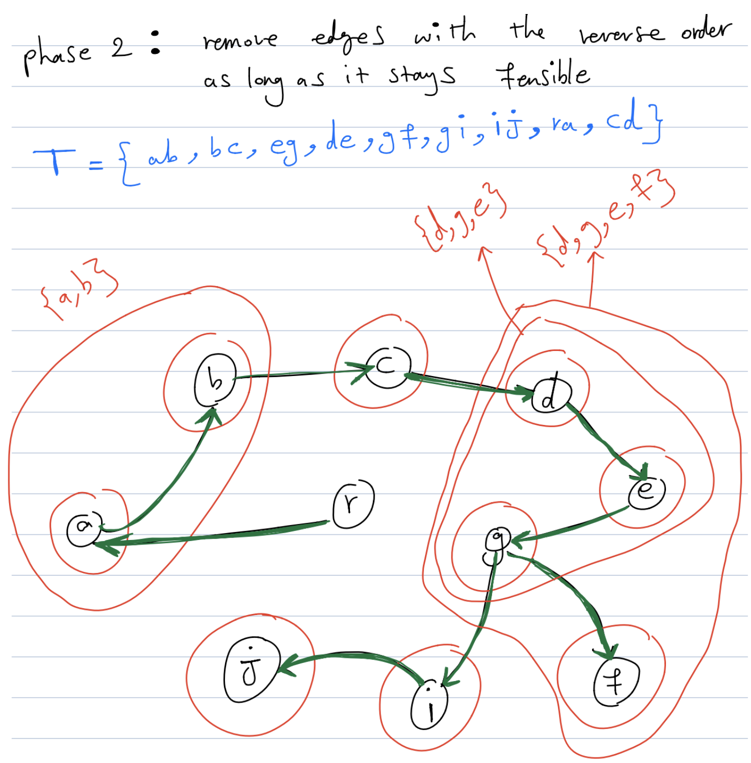


Figure 4: Phase 2