2.2 Shortest Paths

Proposition 2.9 Let y be a feasible potential and let P be a dipath from r to v. Then $c(P) \geq y_v$.

Proof

1. Suppose that $P = v_0, e_1, v_1, \ldots, e_k, v_k$, where $v_0 = r$ and $v_k = v$.
2.

$$\begin{aligned}
c(P) &= \sum_{i=1}^{k} c_{e_i} \geq \sum_{i=1}^{k} (y_{v_i} - y_{v_{i-1}}) \\
&= (\cancel{y}_{v_1} - y_{v_0}) + (\cancel{y}_{v_2} - \cancel{y}_{v_1}) + \cdots + (y_{v_k} - \cancel{y}_{v_{k-1}}) \\
&= y_{v_k} - y_{v_0} \\
&= y_v.
\end{aligned}$$

$\square$

BACK TO SLIDES

Validity of Ford's algorithm

For every $v \in V(G)$, we denote by y_v^j the value of y_v at iteration j, and by $p^j(v)$ be the predecessor of v at iteration j.

Proposition 2.10 (part 1) At every iteration j of Ford's algorithm, if $y_v^j \neq \infty$ then there is a dipath from r to v of cost equal to y_v^j. This dipath is simple if (G, c) has no negative-cost dicircuit.

Proof

1. Let y_v^j be such that $y_v^j \neq \infty$. We show iteratively that y_v^j is the cost of a dipath from r to v.
    - After the initialization of the algorithm this is true.
    - We now assume that this holds at iteration $j-1$ and show it holds at iteration j.
    - If $y_v^j = y_v^{j-1}$ the claim holds by our inductive hypothesis. Suppose that at iteration j we corrected arc ab. We only need to check that the claim is true for node b, since this is the only node whose y value has changed (it has strictly decreased).
    - Since ab has been corrected at iteration j we had $y_b^{j-1} > y_a^{j-1} + c_{ab}$. This implies $y_a^{j-1} < \infty$, and by assumption y_a^{j-1} is the cost of a dipath from r to a.
    - Therefore y_b^j is the cost of the dipath from r to b obtained by appending ab to the dipath from r to a. Note that

this path might be different from the one determined by the current vector of predecessors, see iteration 4 of the example on slide 13.

2. We show that if (G, c) has no negative-cost dicircuit the dipath is simple, see Fig. 1 for an example.

   - Suppose by contradiction that, at iteration j, the dipath to v of cost $y_v^j < \infty$ is not simple. Then, it contains a sequence $v_0, v_1, \ldots, v_k$ of nodes with $v_0 = v_k$.
   - Let q_k be the last iteration $h \leq j$ where y_{v_k} has been decreased. We must have corrected arc $v_{k-1}v_k$.

   $$y_{v_k}^{q_k} = y_{v_{k-1}}^{q_k - 1} + c_{v_{k-1}v_k}.$$

   Similarly, let q_{k-1} be the last iteration $h < q_k$ where $y_{v_{k-1}}$ has been decreased:

   $$y_{v_{k-1}}^{q_{k-1}} = y_{v_{k-1}}^{q_k - 1} \Rightarrow y_{v_k}^{q_k} = y_{v_{k-1}}^{q_{k-1}} + c_{v_{k-1}v_k}.$$

   We must have corrected arc $v_{k-2}v_{k-1}$.

   - In general: Let $q_0 < q_1 < \cdots < q_k \leq j$ be such that q_i is the last iteration $h < q_{i+1}$ where y_{v_i} has been decreased. In this iteration arc $v_{i-1}v_i$ was corrected. Since $y_{v_{i-1}}^{q_i - 1} = y_{v_{i-1}}^{q_{i-1}}$, we have

   $$y_{v_i}^{q_i} = y_{v_{i-1}}^{q_{i-1}} + c_{v_{i-1}v_i} \qquad i = 1, \ldots, k.$$

- The cost of the resulting closed dipath is

$$\begin{aligned}\sum_{i=1}^{k} c_{v_{i-1}v_i} &= \sum_{i=1}^{k}(y_{v_i}^{q_i} - y_{v_{i-1}}^{q_{i-1}}) \\ &= (\cancel{y_{v_1}^{q_1}} - y_{v_0}^{q_0}) + (\cancel{y_{v_2}^{q_2}} - \cancel{y_{v_1}^{q_1}}) + \cdots + (y_{v_k}^{q_k} - \cancel{y_{v_{k-1}}^{q_{k-1}}}) \\ &= y_{v_k}^{q_k} - y_{v_0}^{q_0}.\end{aligned}$$

- But y_{v_k} was lowered at iteration q_k, so this dipath has negative cost, a contradiction, and (i) is proved.

$\square$

The next Lemma is called "Lemma $*$" in the slides. It has *not* been proved in class. We include the proof here for the interested students.

Lemma Consider an iteration j of Ford's algorithm. We have

(i) $y_u^j + c_{uv} \leq y_v^j \quad \forall\, uv \in E : u = p^j(v)$.

Moreover, if at iteration j we have corrected edge $ab \in E$, we have:

(ii) $y_a^j + c_{ab} = y_b^j$ and $a = p^j(b)$.

(iii) $y_b^j + c_{bv} < y_v^j$ for each edge of the form bv with $b = p^j(v)$.

Proof

1. The claim is true at initialization, since no node except r has a predecessor.
2. Suppose the claim is true at iteration $j-1$. We next show that then the claim is true at iteration j. Let $ab \in E$ be the arc that we correct at iteration j.
3. This means that the arc ab was incorrect at iteration $j-1$, i.e.

$$y_b^{j-1} > y_a^{j-1} + c_{ab} = y_b^j. \tag{1}$$

   Thus the value of y_v has strictly decreased at iteration j. Moreover $y_v^j = y_v^{j-1} \forall v \neq b$. In other words, all the components of y stay the same, except for y_b. Also, only only the predecessor of b has changed: $a = p^j(b) \neq p^{j-1}(b)$, and $p^j(v) = p^{j-1}(v)$ for all $v \neq b$.

4. Consider arc ab. We have $a = p^j(b)$ and, since $a \neq b$, we have $y_a^j = y_a^{j-1}$, thus

$$y_b^j = y_a^{j-1} + c_{ab} = y_a^j + c_{ab}.$$

and (ii) holds.
5. Now consider any edge of the form bv such that $p^j(v) = b$. We obtain:

$$y_b^j + c_{bv} < y_b^{j-1} + c_{bv} \leq y_v^{j-1} = y_v^j,$$

where the first inequality follows from (1) (y_b has strictly decreased at iteration j), the second inequality is implied by our inductive hypothesis and $b = p^j(v) = p^{j-1}(v)$, and the equality comes from the fact that only y_b^j has changed in iteration j. This proves (iii).
6. We now prove (i). By our inductive hypothesis we know that

$$y_u^{j-1} + c_{uv} \leq y_v^{j-1} \quad \forall\, uv \in E : u = p^{j-1}(v).$$

If $u \neq b$ and $v \neq b$ we have $y_u^{j-1} = y_u^j$, $y_v^{j-1} = y_v^j$, and $u = p^j(v)$, thus (i) holds in this case.
If $u = b$ (and $v \neq b$), we have $b = p^{j-1}(v) = p^j(v)$, thus (iii) implies that (i) holds.
If $v = b$ we have $a = p^j(b)$, thus (ii) implies that (i) holds.

$\square$

Proposition 2.10 (part 2) Suppose (G, c) has no negative-cost dicircuit. Then at every iteration j of Ford's algorithm, if $p^j(v) \neq -1$, then p^j defines a simple dipath from r to v of cost at most y_v^j.

Proof

1. We show that at any iteration j of Ford's algorithm, if $p^j(v) \neq -1$, then p defines a simple dipath from r to v. The fact that the vector of predecessor defines a dipath ending at v is obvious. We now show that the dipath is simple.

   - If the dipath is not simple, there is a sequence $v_0, v_1, \ldots, v_k$ of nodes with $v_0 = v_k$ and $p^j(v_i) = v_{i-1}$ for $1 \leq i \leq k$.
   - By the previous Lemma, $c_{v_{i-1}v_i} \leq y_{v_i}^j - y_{v_{i-1}}^j$ holds for $i = 1, \ldots, k$, thus the cost of the resulting closed dipath is at most zero:

   $$\sum_{i=1}^{k} c_{v_{i-1}v_i} \leq \sum_{i=1}^{k} (y_{v_i}^j - y_{v_{i-1}}^j) = 0.$$

   - But consider the most recent predecessor assignment on this closed dipath, i.e. the last time that we decreased a component of y for some node $v_1, \ldots, v_k$. Say y_{v_q} was lowered at iteration $h \leq j$, for $q \in \{1, \ldots, k\}$. (Arc $v_{q-1}v_h$ was corrected).

- By the previous Lemma, we have

$$\begin{aligned} c_{v_q v_{q+1}} &< y^h_{v_{q+1}} - y^h_{v_q} = y^j_{v_{q+1}} - y^j_{v_q}, && \text{if } q < k \\ c_{v_0 v_1} &< y^h_{v_1} - y^h_{v_0} = y^j_{v_1} - y^j_{v_0} && \text{if } q = k \end{aligned}$$

- So we have a negative-cost closed dipath

$$\sum_{i=1}^{k} c_{v_{i-1} v_i} < \sum_{i=1}^{k} (y^j_{v_i} - y^j_{v_{i-1}}) = 0.$$

  a contradiction.

2. We have shown that we have a dipath ending at v that is simple. We show that the dipath starts at r.
   - To see this, note that the simple dipath has to start at a node with no predecessor (otherwise there would be a dicircuit). In general, when we set $p(v) = u$, either u has a predecessor, or $u = r$. This shows that the simple dipath starts at r.
3. We show that the simple dipath to v defined by p has cost at most y_v.
   - Let the dipath be $P = v_0, e_1, v_1, \ldots, e_k, v_k$ where $v_0 = r$, $v_k = v$ and $p^j(v_i) = v_{i-1}$ for $1 \leq i \leq k$.
   - $c(P) = \sum_{i=1}^k c_{e_i} \leq \sum_{i=1}^k (y^j_{v_i} - y^j_{v_{i-1}}) = y^j_v - y^j_r = y^j_v$ where the inequality is implied by the previous Lemma and the last equality holds because $y_r = 0$,

□

Theorem 2.11 If (G, c) has no negative-cost dicircuit, then Ford's algorithm terminates after a finite number of iterations. At termination, for each $v \in V$, p defines a least-cost dipath from r to v of cost y_v.

Proof

1. There are finitely many simple dipaths in G.
2. Therefore, by Proposition 2.10(part 1), there are a finite number of possible values for the y_v.
3. Since at each step one of them decreases (and none increases), the algorithm terminates.
4. At termination, for each $v \in V$, y_v is a feasible potential, and, by Proposition 2.10(part 2), p defines a simple dipath from r to v of cost at most y_v.
5. But no dipath to v can have smaller cost than y_v by Proposition 2.9. □

BACK TO SLIDES

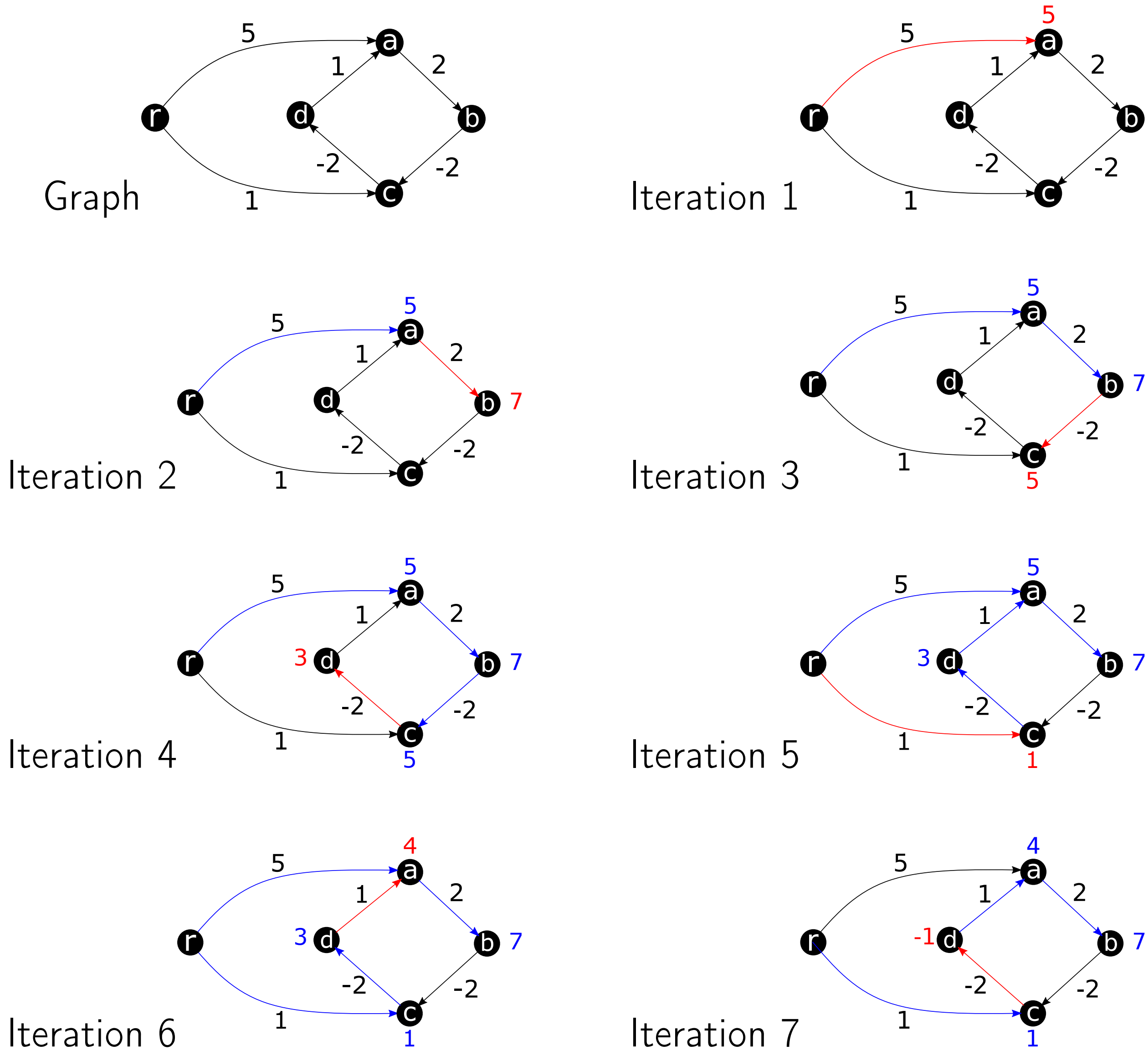

Figure 1: At iteration 7 the dipath to node a of length $y_a^7 = 4$ is r, a, b, c, d, a (note this is not the path given by the predecessors). This dipath is not simple, since it contains the sequence a, b, c, d, a. In the proof, we would consider iteration numbers $2(ab), 3(bc), 4(cd), 6(da)$ respectively. Note that arc cd has been more recently corrected at iteration 7, but iteration 4 is the last iteration before 6 where y_d was decreased.

Acyclic digraphs

Observation G has a topological sort if and only if G is acyclic.

Proof

$\Rightarrow$ If G has a topological sort, then it has no dicircuit. Let $v_1, \ldots, v_n$ be a topological sort of $V(G)$. By contradiction, suppose there is a dicircuit $v_{i_0}, e_1, v_{i_1}, \ldots, e_k, v_{i_k}$ with $e_j = v_{i_{j-1}} v_{i_j}$ for $j = 1, \ldots, k$ and $v_{i_0} = v_{i_k}$. By our assumption we know that $i_{j-1} < i_j$ for all $j = 1, \ldots, k$. This implies $i_0 < i_k$, a contradiction.

$\Leftarrow$ We show that if G is acyclic, then it has a topological sort by induction on $n = |V(G)|$.

1. Base case: $n = 1$.
2. Suppose the claim is true for graphs with n nodes, and prove that the claim is true for graphs with $n + 1$ nodes.
3. Assume G has $n + 1$ nodes. There exists a node v such that there is no $u \in V$ with $uv \in E$. *Why? Prove by contradiction.* We select such node as v_1.
4. Since $G \setminus v$ is acyclic and contains n nodes, by our inductive hypothesis it contains a topological sort $v_1', \ldots, v_n'$.
5. Then $v_1, v_2, \ldots, v_{n+1} = v_1, v_1', \ldots, v_n'$ is a topological sort of G. $\square$

This idea can be turned into an $O(m)$ algorithm to *find a topological sort* (Exercise 2.33).

BACK TO SLIDES

Nonnegative costs

Proposition 2.19 For each $w \in V$, let y'_w be the value of y_w when w is chosen to be scanned. If u is scanned before v, then $y'_u \le y'_v$.

Proof

1. Suppose by contradiction $y'_u > y'_v$ and, without loss of generality, let v be the earliest node scanned after u for which this is true.
2. When u was chosen to be scanned, the value of y_v was at least y'_u. Since later, when v is chosen to be scanned, the value of y_v is smaller than y'_u, we conclude that y_v has been lowered to y'_v after u was chosen to be scanned, but before v was chosen to be scanned.
3. This happened when the algorithm scanned a node w such that $wv \in E$. At this time, y_v was set to $y'_v = y'_w + c_{wv} \ge y'_w$ since $c_{wv} \ge 0$.
4. Thus we obtain $y'_u > y'_v \ge y'_w$, i.e., $y'_u > y'_w$. This contradicts the fact that v is the earliest node scanned after u such that $y'_v > y'_w$. $\square$

Theorem Dijkstra's algorithm is valid.

Proof

1. We show that after all nodes are scanned, we have $y_v + c_{vw} \geq y_w$ for all $vw \in E$.
2. Suppose there exists an arc vw such that, after all nodes are scanned, $y_v + c_{vw} < y_w$. Note that after v was scanned we had $y'_v + c_{vw} \geq y_w$. Thus, it must be that y_v was lowered while another node q was being scanned after v.
3. But then $y_v = y'_q + c_{qv} \geq y'_q$ since $c_{qv} \geq 0$. Since q was scanned later than v, by Proposition 2.19, $y'_q \geq y'_v$. Thus we obtain $y_v \geq y'_v$, a contradiction. $\square$

BACK TO SLIDES

Unit costs

Proposition 2.21 If each $c_e = 1$, then in Dijkstra's algorithm the final value of y_v is the first finite value assigned to it. Moreover, if v is assigned its first finite y_v before q is, then $y_v \leq y_q$.

Proof

1. The statements are true for $v = r$.
2. Suppose the statement is true for all the nodes scanned before v.
3. If $v \neq r$, the first finite value assigned to y_v is $y'_u + 1$, where y'_u is the final value of y_u.
4. By Proposition 2.19, any node w scanned later than u has $y'_w \geq y'_u$, so y_v will not be further decreased. This proves the first statement.
5. To prove the second statement, consider a node q assigned its first finite y_q after v. We have $y_q = y'_w + 1 \geq y'_u + 1 = y_v$. $\square$

BACK TO SLIDES