26.3-3

Suppose that $w(u, v) \geq 0$ for all edges $(u, v) \in E$. What is the relationship between the weight functions w and $\widehat{w}$?

★ 26.4 A general framework for solving path problems in directed graphs

In this section, we examine "closed semirings," an algebraic structure that yields a general framework for solving path problems in directed graphs. We start by defining closed semirings and discussing how they relate to a calculus of directed paths. We then show some examples of closed semirings and a "generic" algorithm for computing all-pairs path information. Both the Floyd-Warshall algorithm and the transitive-closure algorithm from Section 26.2 are instantiations of this generic algorithm.

Definition of closed semirings

A ***closed semiring*** is a system $(S, \oplus, \odot, \overline{0}, \overline{1})$, where S is a set of elements, $\oplus$ (the ***summary operator***) and $\odot$ (the ***extension operator***) are binary operations on S, and $\overline{0}$ and $\overline{1}$ are elements of S, satisfying the following eight properties:

1. $(S, \oplus, \overline{0})$ is a ***monoid***:
   - S is ***closed*** under $\oplus$: $a \oplus b \in S$ for all $a, b \in S$.
   - $\oplus$ is ***associative***: $a \oplus (b \oplus c) = (a \oplus b) \oplus c$ for all $a, b, c \in S$.
   - $\overline{0}$ is an ***identity*** for $\oplus$: $a \oplus \overline{0} = \overline{0} \oplus a = a$ for all $a \in S$.

   Likewise, $(S, \odot, \overline{1})$ is a monoid.
2. $\overline{0}$ is an ***annihilator***: $a \odot \overline{0} = \overline{0} \odot a = \overline{0}$ for all $a \in S$.
3. $\oplus$ is ***commutative***: $a \oplus b = b \oplus a$ for all $a, b \in S$.
4. $\oplus$ is ***idempotent***: $a \oplus a = a$ for all $a \in S$.
5. $\odot$ ***distributes*** over $\oplus$: $a \odot (b \oplus c) = (a \odot b) \oplus (a \odot c)$ and $(b \oplus c) \odot a = (b \odot a) \oplus (c \odot a)$ for all $a, b, c \in S$.
6. If $a_1, a_2, a_3, \ldots$ is a countable sequence of elements of S, then $a_1 \oplus a_2 \oplus a_3 \oplus \cdots$ is well defined and in S.
7. Associativity, commutativity, and idempotence apply to infinite summaries. (Thus, any infinite summary can be rewritten as an infinite summary in which each term of the summary is included just once and the order of evaluation is arbitrary.)
8. $\odot$ distributes over infinite summaries: $a \odot (b_1 \oplus b_2 \oplus b_3 \oplus \cdots) = (a \odot b_1) \oplus (a \odot b_2) \oplus (a \odot b_3) \oplus \cdots$ and $(a_1 \oplus a_2 \oplus a_3 \oplus \cdots) \odot b = (a_1 \odot b) \oplus (a_2 \odot b) \oplus (a_3 \odot b) \oplus \cdots$.

A calculus of paths in directed graphs

Although the closed-semiring properties may seem abstract, they can be related to a calculus of paths in directed graphs. Suppose we are given a directed graph $G = (V, E)$ and a ***labeling function*** $\lambda : V \times V \to S$ mapping all ordered pairs of vertices into some codomain S. The ***label of edge*** $(u, v) \in E$ is denoted $\lambda(u, v)$. Since λ is defined over the domain $V \times V$, the label $\lambda(u, v)$ is usually taken as $\overline{0}$ if (u, v) is not an edge of G (we shall see why in a moment).

We use the associative extension operator $\odot$ to extend the notion of labels to paths. The ***label of path*** $p = \langle v_1, v_2, \ldots, v_k \rangle$ is

$$\lambda(p) = \lambda(v_1, v_2) \odot \lambda(v_2, v_3) \odot \cdots \odot \lambda(v_{k-1}, v_k) .$$

The identity $\overline{1}$ for $\odot$ serves as the label of the empty path.

As a running example of an application of closed semirings, we shall use shortest paths with nonnegative edge weights. The codomain S is $\mathbf{R}^{\geq 0} \cup \{\infty\}$, where $\mathbf{R}^{\geq 0}$ is the set of nonnegative reals, and $\lambda(i, j) = w_{ij}$ for all $i, j \in V$. The extension operator $\odot$ corresponds to the arithmetic operator $+$, and the label of path $p = \langle v_1, v_2, \ldots, v_k \rangle$ is therefore

$$\begin{aligned} \lambda(p) &= \lambda(v_1, v_2) \odot \lambda(v_2, v_3) \odot \cdots \odot \lambda(v_{k-1}, v_k) \\ &= w_{v_1, v_2} + w_{v_2, v_3} + \cdots + w_{v_{k-1}, v_k} \\ &= w(p) . \end{aligned}$$

Not surprisingly, the role of $\overline{1}$, the identity for $\odot$, is taken by 0, the identity for $+$. We denote the empty path by ε, and its label is $\lambda(\varepsilon) = w(\varepsilon) = 0 = \overline{1}$.

Because the extension operator $\odot$ is associative, we can define the label of the concatenation of two paths in a natural way. Given paths $p_1 = \langle v_1, v_2, \ldots, v_k \rangle$ and $p_2 = \langle v_k, v_{k+1}, \ldots, v_l \rangle$, their ***concatenation*** is

$$p_1 \circ p_2 = \langle v_1, v_2, \ldots, v_k, v_{k+1}, \ldots, v_l \rangle ,$$

and the label of their concatenation is

$$\begin{aligned} \lambda(p_1 \circ p_2) &= \lambda(v_1, v_2) \odot \lambda(v_2, v_3) \odot \cdots \odot \lambda(v_{k-1}, v_k) \odot \\ &\qquad \lambda(v_k, v_{k+1}) \odot \lambda(v_{k+1}, v_{k+2}) \odot \cdots \odot \lambda(v_{l-1}, v_l) \\ &= (\lambda(v_1, v_2) \odot \lambda(v_2, v_3) \odot \cdots \odot \lambda(v_{k-1}, v_k)) \odot \\ &\qquad (\lambda(v_k, v_{k+1}) \odot \lambda(v_{k+1}, v_{k+2}) \odot \cdots \odot \lambda(v_{l-1}, v_l)) \\ &= \lambda(p_1) \odot \lambda(p_2) . \end{aligned}$$

The summary operator $\oplus$, which is both commutative and associative, is used to ***summarize*** path labels. That is, the value $\lambda(p_1) \oplus \lambda(p_2)$ gives a summary, the semantics of which are specific to the application, of the labels of paths p_1 and p_2.

Our goal will be to compute, for all pairs of vertices $i, j \in V$, the summary of all path labels from i to j:

$$l_{ij} = \bigoplus_{i \overset{p}{\leadsto} j} \lambda(p) \ . \tag{26.11}$$

We require commutativity and associativity of $\oplus$ because the order in which paths are summarized should not matter. Because we use the annihilator $\overline{0}$ as the label of an ordered pair (u, v) that is not an edge in the graph, any path that attempts to take an absent edge has label $\overline{0}$.

For shortest paths, we use min as the summary operator $\oplus$. The identity for min is ∞, and ∞ is indeed an annihilator for $+$: $a + \infty = \infty + a = \infty$ for all $a \in \mathbf{R}^{\geq 0} \cup \{\infty\}$. Absent edges have weight ∞, and if any edge of a path has weight ∞, so does the path.

We want the summary operator $\oplus$ to be idempotent, because from equation (26.11), we see that $\oplus$ should summarize the labels of a set of paths. If p is a path, then $\{p\} \cup \{p\} = \{p\}$; if we summarize path p with itself, the resulting label should be the label of p: $\lambda(p) \oplus \lambda(p) = \lambda(p)$.

Because we consider paths that may not be simple, there may be a countably infinite number of paths in a graph. (Each path, simple or not, has a finite number of edges.) The operator $\oplus$ should therefore be applicable to a countably infinite number of path labels. That is, if $a_1, a_2, a_3, \ldots$ is a countable sequence of elements in codomain S, then the label $a_1 \oplus a_2 \oplus a_3 \oplus \cdots$ should be well defined and in S. It should not matter in which order we summarize path labels, and thus associativity and commutativity should hold for infinite summaries. Furthermore, if we summarize the same path label a a countably infinite number of times, we should get a as the result, and thus idempotence should hold for infinite summaries.

Returning to the shortest-paths example, we ask if min is applicable to an infinite sequence of values in $\mathbf{R}^{\geq 0} \cup \{\infty\}$. For example, is the value of $\min_{k=1}^{\infty} \{1/k\}$ well defined? It is, if we think of the min operator as actually returning the greatest lower bound (infimum) of its arguments, in which case we get $\min_{k=1}^{\infty} \{1/k\} = 0$.

To compute labels of diverging paths, we need distributivity of the extension operator $\odot$ over the summary operator $\oplus$. As shown in Figure 26.7, suppose that we have paths $u \overset{p_1}{\leadsto} v$, $v \overset{p_2}{\leadsto} x$, and $v \overset{p_3}{\leadsto} y$. By distributivity, we can summarize the labels of paths $p_1 \circ p_2$ and $p_1 \circ p_3$ by computing either $(\lambda(p_1) \odot \lambda(p_2)) \oplus (\lambda(p_1) \odot \lambda(p_3))$ or $\lambda(p_1) \odot (\lambda(p_2) \oplus \lambda(p_3))$.

Because there may be a countably infinite number of paths in a graph, $\odot$ should distribute over infinite summaries as well as finite ones. Figure 26.8, for example, contains paths $u \overset{p_1}{\leadsto} v$ and $v \overset{p_2}{\leadsto} x$, along with the cycle $v \overset{c}{\leadsto} v$. We must be able to summarize the paths $p_1 \circ p_2$, $p_1 \circ c \circ p_2$, $p_1 \circ c \circ c \circ p_2$, ... Distributivity of $\odot$ over countably infinite summaries gives us

$$\begin{aligned}
&(\lambda(p_1) \odot \lambda(p_2)) \oplus (\lambda(p_1) \odot \lambda(c) \odot \lambda(p_2)) \\
&\qquad \oplus (\lambda(p_1) \odot \lambda(c) \odot \lambda(c) \odot \lambda(p_2)) \oplus \cdots \\
&\quad = \lambda(p_1) \odot (\lambda(p_2) \oplus (\lambda(c) \odot \lambda(p_2)) \oplus (\lambda(c) \odot \lambda(c) \odot \lambda(p_2)) \oplus \cdots) \\
&\quad = \lambda(p_1) \odot (\overline{1} \oplus c \oplus (c \odot c) \oplus (c \odot c \odot c) \oplus \cdots) \odot \lambda(p_2) \ .
\end{aligned}$$

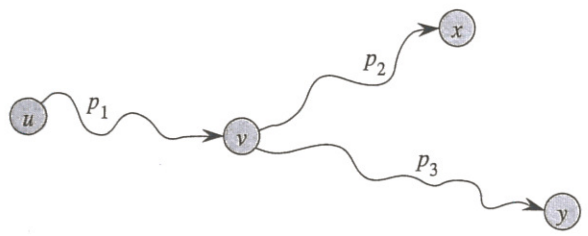

Figure 26.7 Using distributivity of $\odot$ over $\oplus$. To summarize the labels of paths $p_1 \circ p_2$ and $p_1 \circ p_3$, we may compute either $(\lambda(p_1) \odot \lambda(p_2)) \oplus (\lambda(p_1) \odot \lambda(p_3))$ or $\lambda(p_1) \odot (\lambda(p_2) \oplus \lambda(p_3))$.

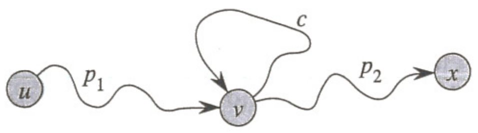

Figure 26.8 Distributivity of $\odot$ over countably infinite summaries of $\oplus$. Because of cycle c, there are a countably infinite number of paths from vertex v to vertex x. We must be able to summarize the paths $p_1 \circ p_2$, $p_1 \circ c \circ p_2$, $p_1 \circ c \circ c \circ p_2$, ...

We use a special notation to denote the label of a cycle that may be traversed any number of times. Suppose that we have a cycle c with label $\lambda(c) = a$. We may traverse c zero times for a label of $\lambda(\varepsilon) = \overline{1}$, once for a label of $\lambda(c) = a$, twice for a label of $\lambda(c) \odot \lambda(c) = a \odot a$, and so on. The label we get by summarizing this infinite number of traversals of cycle c is the ***closure*** of a, defined by

$$a^* = \overline{1} \oplus a \oplus (a \odot a) \oplus (a \odot a \odot a) \oplus (a \odot a \odot a \odot a) \oplus \cdots .$$

Thus, in Figure 26.8, we want to compute $\lambda(p_1) \odot (\lambda(c))^* \odot \lambda(p_2)$.

For the shortest-paths example, for any nonnegative real $a \in \mathbf{R}^{\geq 0} \cup \{\infty\}$,

$$\begin{aligned} a^* &= \min_{k=0}^{\infty} \{ka\} \\ &= 0 . \end{aligned}$$

The interpretation of this property is that since all cycles have nonnegative weight, no shortest path ever needs to traverse an entire cycle.

Examples of closed semirings

We have already seen one example of a closed semiring, namely $S_1 = (\mathbf{R}^{\geq 0} \cup \{\infty\}, \min, +, \infty, 0)$, which we used for shortest paths with nonnegative edge weights. (As previously noted, the min operator actually returns the greatest lower bound of its arguments.) We have also shown that $a^* = 0$ for all $a \in \mathbf{R}^{\geq 0} \cup \{\infty\}$.

We claimed, however, that even if there are negative-weight edges, the Floyd-Warshall algorithm computes shortest-path weights as long as no negative-weight cycles are present. By adding the appropriate closure operator and extending the codomain of labels to $\mathbf{R} \cup \{-\infty, +\infty\}$, we can find a closed semiring to handle negative-weight cycles. Using min for $\oplus$ and $+$ for $\odot$, the reader may verify that the closure of $a \in \mathbf{R} \cup \{-\infty, +\infty\}$ is

$$a^* = \begin{cases} 0 & \text{if } a \geq 0 , \\ -\infty & \text{if } a < 0 . \end{cases}$$

The second case ($a < 0$) models the situation in which we can traverse a negative-weight cycle an infinite number of times to obtain a weight of $-\infty$ on any path containing the cycle. Thus, the closed semiring to use for the Floyd-Warshall algorithm with negative edge weights is $S_2 = (\mathbf{R} \cup \{-\infty, +\infty\}, \min, +, +\infty, 0)$. (See Exercise 26.4-3.)

For transitive closure, we use the closed semiring $S_3 = (\{0, 1\}, \vee, \wedge, 0, 1)$, where $\lambda(i, j) = 1$ if $(i, j) \in E$, and $\lambda(i, j) = 0$ otherwise. Here we have $0^* = 1^* = 1$.

A dynamic-programming algorithm for directed-path labels

Suppose we are given a directed graph $G = (V, E)$ with labeling function $\lambda : V \times V \to S$. The vertices are numbered 1 through n. For each pair of vertices $i, j \in V$, we want to compute equation (26.11):

$$l_{ij} = \bigoplus_{i \overset{p}{\leadsto} j} \lambda(p) ,$$

which is the result of summarizing all paths from i to j using the summary operator $\oplus$. For shortest paths, for example, we wish to compute

$$l_{ij} = \delta(i, j) = \min_{i \overset{p}{\leadsto} j} \{w(p)\} .$$

There is a dynamic-programming algorithm to solve this problem, and its form is very similar to the Floyd-Warshall algorithm and the transitive-closure algorithm. Let $Q_{ij}^{(k)}$ be the set of paths from vertex i to vertex j with all intermediate vertices in the set $\{1, 2, \ldots, k\}$. We define

$$l_{ij}^{(k)} = \bigoplus_{p \in Q_{ij}^{(k)}} \lambda(p) .$$

Note the analogy to the definitions of $d_{ij}^{(k)}$ in the Floyd-Warshall algorithm and $t_{ij}^{(k)}$ in the transitive-closure algorithm. We can define $l_{ij}^{(k)}$ recursively by

$$l_{ij}^{(k)} = l_{ij}^{(k-1)} \oplus \left(l_{ik}^{(k-1)} \odot (l_{kk}^{(k-1)})^* \odot l_{kj}^{(k-1)} \right) . \qquad (26.12)$$

Recurrence (26.12) is reminiscent of recurrences (26.5) and (26.8), but with an additional factor of $(l_{kk}^{(k-1)})^*$ included. This factor represents the summary of all cycles that pass through vertex k and have all other vertices in the set $\{1, 2, \ldots, k-1\}$. (When we assume no negative-weight cycles in the Floyd-Warshall algorithm, $(c_{kk}^{(k-1)})^*$ is 0, corresponding to $\overline{1}$, the weight of an empty cycle. In the transitive-closure algorithm, the empty path from k to k gives us $(t_{kk}^{(k-1)})^* = 1 = \overline{1}$. Thus, for both of these algorithms, we can ignore the factor of $(l_{kk}^{(k-1)})^*$, since it is just the identity for $\odot$.) The basis of the recursive definition is

$$l_{ij}^{(0)} = \begin{cases} \lambda(i,j) & \text{if } i \neq j , \\ \overline{1} \oplus \lambda(i,j) & \text{if } i = j , \end{cases}$$

which we can see as follows. The label of the one-edge path $\langle i, j \rangle$ is simply $\lambda(i,j)$ (which is equal to $\overline{0}$ if (i,j) is not an edge in E). If, in addition, $i = j$, then $\overline{1}$ is the label of the empty path from i to i.

The dynamic-programming algorithm computes the values $l_{ij}^{(k)}$ in order of increasing k. It returns the matrix $L^{(n)} = \left(l_{ij}^{(n)}\right)$.

```
COMPUTE-SUMMARIES($\lambda$, $V$)
 1  $n \leftarrow |V|$
 2  for $i \leftarrow 1$ to $n$
 3      do for $j \leftarrow 1$ to $n$
 4             do if $i = j$
 5                   then $l_{ij}^{(0)} \leftarrow \overline{1} \oplus \lambda(i,j)$
 6                   else $l_{ij}^{(0)} \leftarrow \lambda(i,j)$
 7  for $k \leftarrow 1$ to $n$
 8      do for $i \leftarrow 1$ to $n$
 9             do for $j \leftarrow 1$ to $n$
10                    do $l_{ij}^{(k)} \leftarrow l_{ij}^{(k-1)} \oplus \left(l_{ik}^{(k-1)} \odot (l_{kk}^{(k-1)})^* \odot l_{kj}^{(k-1)}\right)$
11  return $L^{(n)}$
```

The running time of this algorithm depends on the time to compute $\odot$, $\oplus$, and $*$. If we let $T_{\odot}$, $T_{\oplus}$, and T_* represent these times, then the running time of COMPUTE-SUMMARIES is $\Theta(n^3(T_{\odot} + T_{\oplus} + T_*))$, which is $\Theta(n^3)$ if each of the three operations takes $O(1)$ time.

Exercises

26.4-1
Verify that $S_1 = (\mathbf{R}^{\geq 0} \cup \{\infty\}, \min, +, \infty, 0)$ and $S_3 = (\{0,1\}, \vee, \wedge, 0, 1)$ are closed semirings.

26.4-2
Verify that $S_2 = (\mathbf{R} \cup \{-\infty, +\infty\}, \min, +, +\infty, 0)$ is a closed semiring. What is the value of $a + (-\infty)$ for $a \in \mathbf{R}$? What about $(-\infty) + (+\infty)$?