Notes on Semirings and Generalized Path Computations

Chris Dyer

October 8, 2015

1 Introduction

We have have considered the Viterbi and Forward algorithms, which compute the **maximum** or the **sum** (respectively) over all paths in a WFSA (e.g., in an HMM trellis). These are dynamic programming algorithms that do not need to explicitly enumerate all paths in the machine to compute these **path sums**; rather, they exploit the fact that in a WFSA, many paths will share substructure and the maximum or sum associated with this substructure can be computed just once.

Today's lecture has two goals:

1. We generalize the notion of **weights** to make precise the conditions under which the Viterbi/Forward algorithm can be used to compute the **path sum** under other operations.

2. We provide an algorithm for computing path sums of WFSAs that contain loops.

When can we use the Viterbi/Forward algorithm? Path weight aggregation computation can be computed efficiently with just one algorithm provided that the weights, the aggregation function, and how path component weights "multiply" have certain properties. Namely, these must form a **semiring**. A semiring is an algebraic structure consisting of:

- $\mathbb{K}$, a set (e.g., the real numbers, the natural numbers, $\mathbb{R} \cup \{-\infty, +\infty\}, \ldots$);

- $\oplus$, an addition operator that is associative and commutative (i.e., for all $a, b, c \in \mathbb{K}^3$ associativity implies $(a \oplus b) \oplus c = a \oplus (b \oplus c)$ and commutativity implies $a \oplus b = b \oplus a$);

- $\otimes$, a multiplication operator that is associative (i.e., for all $a, b, c \in \mathbb{K}$, $(a \otimes b) \otimes c = a \otimes (b \otimes c)$);

- $\overline{0} \in \mathbb{K}$, an additive identity (i.e., $\overline{0} + a = a$ for all $a \in \mathbb{K}$) that is also an "annihilator," i.e., $\overline{0} \otimes a = \overline{0}$; and

- $\overline{1} \in \mathbb{K}$, a multiplicative identity; i.e., $\overline{1} \otimes a = a$ for all $a \in \mathbb{K}$.

Additionally, $\otimes$ must distribute over $\oplus$, that is $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$ and $(b \oplus c) \otimes a = (b \otimes a) \oplus (c \otimes a)$. If $\otimes$ is commutative the semiring is said to be *commutative*. If $\oplus$ is idempotent (i.e., for every $a \in \mathbb{K}$, $a \oplus a = a$), then the semiring is said to be *idempotent*. A semiring may have an additional *closure operator* a^* which satisfies the axiom $a^* = \overline{1} \oplus a \otimes a^*$. A semiring equipped with a closure operator is said to be *closed*.

A note on closure. There are generally two ways to think about the definition of the closure operator:

- In semirings where infinite series are well defined, $a^* = \overline{1} \oplus a \oplus a^2 \oplus a^3 \oplus \cdots$;
- in semirings where $\oplus$ has an inverse operation (subtraction) and reciprocals are defined, it is possible to define closure as $a^* = (\overline{1} \ominus a)^{-1}$.

Both of these definitions fulfill the $a^* = \overline{1} \oplus a \otimes a^*$ axiom.

1.1 Example: tropical semiring

$(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$ is a semiring.

1.2 Example: counting semiring

$(\mathbb{N} \cup \{+\infty\}, +, \times, 0, 1)$ is a semiring.

1.3 Example: Boolean semiring

$(\{\text{TRUE}, \text{FALSE}\}, \vee, \wedge, \text{FALSE}, \text{TRUE})$ is a semiring.

1.4 Example: probability semiring

$(\mathbb{R}_{\geq 0} \cup \{+\infty\}, +, \times, 0, 1)$ is a semiring.

2 Generalized path sums

Using the definition of semirings, can now generalize our definition of weights assigned to paths as:

$$w(\pi) = \lambda(p(\pi)) \otimes \left(\bigotimes_{e \in \pi} w(e) \right) \otimes \rho(n(\pi))$$

When we wish to aggregate weights over a set of paths P (for example, all the possible paths producing a string $\boldsymbol{s} \in \Sigma^*$), we now have

$$\begin{aligned} w(P) &= \bigoplus_{\pi \in P} w(\pi) \\ &= \bigoplus_{\pi \in P} \left(\lambda(p(\pi)) \otimes \left(\bigotimes_{e \in \pi} w(e) \right) \otimes \rho(n(\pi)) \right) \end{aligned}$$

3 Example question

Consider the following definitions:

$$\begin{aligned} \mathbb{K} &= \mathbb{R}_{\geq 0} \times 2^Q \\ (a, A) \otimes (b, B) &= (a \times b, A \cup B) \\ \overline{1} &= (1, \emptyset) \\ (a, A) \oplus (b, B) &= \begin{cases} (a, A) & \text{if } a > b \\ (a, A \cup B) & \text{if } a = b \\ (b, B) & \text{if } a < b \end{cases} \\ \overline{0} &= (0, \emptyset) \end{aligned}$$

Is this a semiring? **No**. For some $a \in \mathbb{K}$, $a \otimes \overline{0} \neq \overline{0}$. How can this be fixed? Assume we have a WFSA with weights $w : E \mapsto \mathbb{R}_{\geq 0}$, and starting and ending weights 1. Redefine the weight function such that $w'(e) = (w(e), p(e))$, what quantity does the path-sum correspond to?

4 The Forward algorithm (computes path-sum for acyclic WFSAs)

Assuming we have a semiring-weighted WFSA $\mathcal{M} = (Q, \Sigma, I, F, E, \lambda, \rho, w)$ with no ε transitions and a single initial state q_0 and a single final state q_f, the following algorithm computes the $\oplus$-sum of all paths from q_0 from q_f in time $O(|E|)$:

```
for all q ∈ Q − {q0} do
    d(q) ← 0̄
end for
d(q0) ← λ(q0)
perform a topological sort on Q
for all q ∈ Q in top-sort order do
    for all (q, x, r) ∈ E do
        d(r) ← d(r) ⊕ d(q) ⊗ w(q, x, r)
    end for
end for
return d(qf) ⊗ ρ(qf)
```

5 Path expressions and computing the path-sum

Assume we have a semiring-weighted WFSA $\mathcal{M} = (Q, \Sigma, I, F, E, \lambda, \rho, w)$. For simplicity, further assume that $I = \{q_0\}$, $F = \{q_f\}$, $\lambda(q_0) = \overline{1}$, and $\rho(q_f) = \overline{1}$.[1] Recall that a **regular expression** is a description of the languages accepted by an FSA that is defined inductively:

- $\emptyset$ is a regular expression;
- ε is a regular expression;
- for each $a \in \Sigma$, a is a regular expression denoting $\{a\}$;
- if r and s are regular expressions denoting, respectively, the languages R and S, then
  - $(r \mid s)$ denotes $R \cup S$;
  - (rs) denotes $R \cdot S$; and
  - r^* denotes R^*.

Recall that there is an algorithm for converting any deterministic FSA into a regular expression.

5.1 Path expressions

A **path expression** $P(q, r)$ is a regular expression whose language $L(P(q, r))$ is the (possibly infinite) set of path strings $\subseteq E^*$ leading from state $q \in Q$ to state $r \in Q$. We may construct this regular expression for any $\mathcal{M}$ by recalling that any deterministic FSA can be converted into an regular expression. Although $\mathcal{M}$ will not in general be deterministic, we will convert it into a **path automaton** $\mathcal{M}_\pi$ by setting $\Sigma_\pi = E$ and letting each edge $e \in E_\pi$ be labeled with the corresponding edge $(q, \sigma, r) \in E$. This machine is trivially

[1] *Any* WFSA can be converted to this form by adding appropriately weighted ε-transitions.

deterministic since every edge has a unique label, and therefore it can be converted into a regular expression representing $P(q, r)$.

5.2 Path-sums from regular expressions

We are now in a position to define an algorithm that computes the $\oplus$-aggregation of all (possibly infinitely many!) paths in $P(q, r)$, that is:

$$w(P(q,r)) = \bigoplus_{\pi \in L(P(q,r))} \bigotimes_{i=1}^{|\pi|} w(e_i)$$

We define w inductively as follows:

- $w(\emptyset) = \overline{0}$;
- $w(\varepsilon) = \overline{1}$;
- for each $e \in E$ $(= \Sigma_\pi)$, $w(e)$ is defined according to $\mathcal{M}$;
- if r and s are path expressions with weights $w(r)$ and $w(s)$, then
  - $w(r \mid s) = w(r) \oplus w(s)$;
  - $w(rs) = w(r) \otimes w(s)$; and
  - $w(r^*) = w(r)^*$.

Thus, computing the $\oplus$-aggregation of all paths in a machine $\mathcal{M}$ can be accomplished by forming the path automaton $\mathcal{M}_\pi$ and converting this into a regular expression.