LECTURE - 4

Recall : Given an weighted automaton $\mathcal{A}$, Support $(\llbracket \mathcal{A} \rrbracket) = \{w \mid \llbracket \mathcal{A} \rrbracket (w) \neq 0\}$

Non-emptiness Problem for Support Languages :

Input : Weighted Automaton $\mathcal{A}$ over $\mathbb{S}$ and Σ

Question : Is Support $(\llbracket \mathcal{A} \rrbracket) \neq \phi$?

We return to the matrix representation for $\mathcal{A} = \langle \lambda, \mu, \gamma \rangle$

λ - $1 \times n$ row vector $\in \mathbb{S}^n$

μ - $\Sigma \longrightarrow$ ($n \times n$ ~~rows~~ matrix over $\mathbb{S}$)

γ - $n \times 1$ column vector

Can think of μ_a for $a \in \Sigma$ as a weight distortion.

With each word, the automaton $\mathcal{A}$ associates a vector in $\mathbb{S}^n$, namely the vector $\lambda \cdot \mu(w)$ with the word w. Here, $\mu(w)$ is defined as in LECTURE-2.

For the sake of ~~simpl~~ the results that will follow, we will assume that $\mathbb{S}$ is a field, i.e, inverses of $+$ and $\times$ exist in $\mathbb{S}$.

Define the set of <u>Reachable Vectors</u> in $\mathbb{S}^n$ as follows :

$$\lambda \mu(\Sigma^*) = \left\{ \lambda \mu(w) \;\middle|\; w \in \Sigma^* \right\} \subseteq \mathbb{S}^n$$

We are intersted in the span of reachable vectors in $\mathbb{S}^n$, denoted $\langle \lambda \cdot \mu(\Sigma^*) \rangle$.
This is a subspace of $\mathbb{S}^n$ and we may compute its basis.

The following algorithm computes ~~the~~ a basis for $\langle \lambda \mu (\Sigma^*) \rangle$.

$B \longleftarrow \{\lambda\}$

Todo $\longleftarrow \{\lambda\}$

while Todo $\neq \phi$:

remove x from Todo

If $x \cdot \mu(a) \notin \langle B \rangle$,

add $x \mu(a)$ to B

add $x \mu(a)$ to Todo

If $x \cdot \mu(b) \notin \langle B \rangle$,

add $x \mu(b)$ to B

add $x \mu(b)$ to Todo

Output B

Now, we argue the correctness of the algorithm above.

- We observe that the set B is always linearly independent. This can be established as a loop invariant. This proves that B would be linearly independent if the algorithm halts

- Since B is always linearly independent, there may be at most n additions to Todo. After another n iterations, Todo must be empty. Therefore, the algorithm terminates.

- Suppose the algorithm returns $B = \{x_1 \dots x_k\}$. We argue, by induction, that $\lambda \mu(w) \in \langle B \rangle$, for all $w \in \Sigma^*$

+ Base Case $\lambda \mu(\varepsilon) = \lambda$ is added to B manually in the beginning

+ Inductive Step Suppose $\lambda \mu(w) \in \langle B \rangle$ and in particular $\lambda \cdot \mu(w) = \sum_{i=1}^{k} \alpha_i x_i$ for $\alpha_i \in S$.

$$(\lambda \cdot \mu(w)) \cdot \mu(a) = \sum_{i=1}^{k} \alpha_i x_i \cdot \mu(a)$$

But the algorithm ~~is such~~ maintains that $x_i = \lambda\mu(w_i)$ for some $|w_i| < n$

So, $(\lambda\mu(w))\cdot\mu(a) = \sum_{i=1}^{k} \alpha_i (x_i\, \mu(a))$

$= \sum_{i=1}^{k} \alpha_i \lambda\mu(w_i a)$ ——— (1)

Now, either $\lambda\mu(w_i a) \in B$, in which case we are done. Otherwise $\lambda\mu(w_i a) \notin B$. But we know that $\lambda\mu(w_i) \in B$. So, $\lambda\mu(w_i)$ was an element of Todo at some point. But $\lambda\mu(w_i a)$ was not added to B. This means $\lambda\mu(w_i a) = \sum_{i=1}^{k} \beta_i x_i$

So, (1) can be rewritten as $\sum_{i=1}^{k} \alpha_i \left(\sum_{i=1}^{k} \beta_i x_i\right)$

but this sum can be expressed as a ~~sum~~ linear combination of elements in B.

A similar argument holds for $\lambda\mu(wb)$.

QED

Checking whether a given vector is in the span of B can be done using Gaussian Elimination which requires $O(n^3)$ operations.

Thus, the algorithm terminates in $O(|\Sigma| \times n \times n^3)$ time.

Claim Suppose $B = \{x_1 \ldots x_k\}$ is a basis of $\langle \lambda\mu(\Sigma^*)\rangle$ obtained by the algorithm above. Then Support $([\![\langle \lambda, \mu, \gamma\rangle]\!])$ is empty iff $x_i \cdot \gamma = 0$ for each i.

Proof ($\Leftarrow$) Given any word $w \in \Sigma^*$, $\lambda\mu(w)$ can be realised as a linear combination of B, say $\sum_{i=1}^{k} x_i \alpha_i$. Therefore, $\lambda\mu(w)\gamma$ is $\sum \alpha_i x_i \gamma_i$

$= \sum \alpha_i (x_i \cdot \gamma_i) = \sum \alpha_i \cdot 0 = 0$

Thus $w \notin Supp([\![\langle \lambda, \mu, \gamma\rangle]\!])$

($\Longrightarrow$) If there is some ~~word~~ x_i s.t $x_i \cdot \gamma \neq 0$, there is a corresponding w_i s.t $\lambda \mu(w_i) = x_i$ and hence $\lambda \mu(w_i) \gamma \neq 0$. Thus, $w_i \in Supp(⟦\langle \lambda, \mu, \gamma \rangle⟧)$

QED

So, we have a polynomial time algorithm for checking the non-emptiness of the support language.

Equivalence Problem for weighted Automata

Input: $\mathcal{A}_1$, $\mathcal{A}_2$ - weighted automata

Question: Is $⟦A_1⟧ = ⟦A_2⟧$?

Observe that if $\mathcal{A}_1 = \langle \lambda_1, \mu_1, \gamma_1 \rangle$ and $\mathcal{A}_2 = \langle \lambda_2, \mu_2 \gamma_2 \rangle$, we may consider $\mathcal{A} = \langle \lambda, \mu, \gamma \rangle$ with

$$\lambda = [\overbrace{\cdots}^{\lambda_1} \quad \underbrace{\cdots}_{-\lambda_2}]$$

$$\mu(a) = \begin{bmatrix} \mu_1(a) & 0 \\ 0 & \mu_2(a) \end{bmatrix} \quad \text{for } a \in \Sigma$$

$$\gamma = \begin{bmatrix} \gamma_1 \\ \\ \gamma_2 \end{bmatrix}$$

This is simply the disjoint union of $\mathcal{A}_1$ and $\mathcal{A}_2$ with the negated initial weights for $\mathcal{A}_2$. $⟦A⟧$ then computes $⟦\mathcal{A}_1⟧ - ⟦\mathcal{A}_2⟧$.

Thus, to answer the equivalence problem, we may construct $\mathcal{A}$ as above and check if Support $(⟦\mathcal{A}⟧) = \phi$

Exercise Assume that $\mathbb{S}$ is an ordered field. We say that $⟦\mathcal{A}_1⟧ \leqslant ⟦\mathcal{A}_2⟧$ if $⟦\mathcal{A}_1⟧(w) \leqslant ⟦\mathcal{A}_2⟧(w)$, read as, $\mathcal{A}_1$ is dominated by $\mathcal{A}_2$. Given $\mathcal{A}_1$ and $\mathcal{A}_2$, ~~for~~ check algorithmically if $\mathcal{A}_1$ dominates $\mathcal{A}_2$.

Remark The arguments above hold if $\mathbb{S}$ is embedded in some field.
For example, $\mathbb{N}$ can be embedded in $\mathbb{Q}$ and so can be $\mathbb{Z}$.
In general, this may not be possible. For example, consider $(\mathbb{N} \cup \{\infty\}, \min, +)$
Suppose we could embed this in a ring with elements n' for each $n \in \mathbb{N}$ s.t. $\min(n, n') = \infty$. Then, we could get a contradiction as follows:

$$\min(1, 2) = \min(1, 3)$$
$$\Rightarrow \min(1', \min(1, 2)) = \min(1', \min(1, 3))$$
$$\Rightarrow \min(\min(1', 1), 2) = \min(\min(1', 1), 3)$$
$$\Rightarrow \min(\infty, 2) = \min(\infty, 3)$$
$$\Rightarrow 2 = 3$$

Thus, in general, ~~there~~ cancellativity is necessary for a semiring to be embedded in a field. In the semiring $(\{0, 1\}, \vee, \wedge)$, ~~there~~ we have $0 \vee 1 = 1 \vee 1 = 1$
~~Hence~~ Hence, these arguments do not hold. Indeed, checking equivalence for NFA is PSPACE-complete.

However, if a semiring can be embedded in a ring without any zero-divisors, (i.e, the ring maintains $m \cdot n = 0 \Rightarrow m = 0 \vee n = 0$), then it may be embedded in its corresponding field of fractions.