Progamming Contest

July 29, 2001

The basic idea is that the input document gives us an upper bound on the number of elements n_E that may occur in the output document.

An input document consists of a finite set C of data items. These data items are numbered by successive integers: we capture this by a function $\pi : C \to \mathbb{N}$ which is a bijection between C and $\{1 \ \ldots \ n_C\}$ where $n_C = |C|$. Furthermore, we assume given a set A of attributes and an assignment $\alpha_C : C \times A \to \mathbb{N}$.

We assume given a set T of tags and we define the cost of a tag by a function $\mathsf{cost}_T : T \to \mathbb{N}$:

`<b> ... </b>`	$\mathsf{cost}_T(\mathtt{b}) = 7$
`<pl> ... </pl>`	$\mathsf{cost}_T(\mathtt{pl}) = 9$

an upper bound for the number of element is:

$$\lfloor \Sigma\{\mathsf{cost}_T(e) \mid e \text{ in document}\}/7 \rfloor$$

it may be possible to derive a tighter bound when e.g. the input document contains redundant elements etc...

Our formalization of the output document considers the set C of data items ($|C| = n_C$) and a set E of elements ($|E| = n_E$). We write $V = E \uplus C$ for the set of nodes in the tree representation of the document. We also assume an additional node root to serve as the root node of the document. We write $V^* = V \uplus \mathsf{root}$. Not all elements in E will necessarily be used (hopefully not): in our formalization, these unused elements will be empty and immediate daughters of the root. We consider the following functions:

$$\begin{aligned} \mathsf{mother} &: V \to V^* \\ \mathsf{daughters} &: V^* \to 2^V \\ \mathsf{down} &: V^* \to 2^V \\ \mathsf{eqdown} &: V^* \to 2^{V^*} \end{aligned}$$

The treeness condition is satisfied when $\forall w \in V^*$ and $\forall w' \in V$:

$$\begin{aligned}
\mathsf{eqdown}(w) &= \{w\} \uplus \mathsf{down}(w)\\
\mathsf{down}(w) &= \cup\{\mathsf{eqdown}(w') \mid w' \in \mathsf{daughters}(w)\}\\
w = \mathsf{mother}(w') \ &\equiv\ w' \in \mathsf{daughters}(w)\\
V &= \uplus\{\mathsf{daughters}(w) \mid w \in V^*\}
\end{aligned}$$

Posing $V = \{w_1, \ldots, w_n\}$, equation $\mathsf{down}(w) = \cup\{\mathsf{eqdown}(w') \mid w' \in \mathsf{daughters}(w)\}$ can be implemented by the *selection union* constraint:

$$\mathsf{down}(w) = \cup\langle \mathsf{eqdown}(w_1), \ldots, \mathsf{eqdown}(w_n)\rangle[\mathsf{daughters}(w)]$$

We introduce an additional fictional tag ϵ to be assigned to an element which is not used. We write $T^* = T \uplus \{\epsilon\}$. We introduce the function $\tau : E \to T^*$ to represent an assignment of tags to elements.

Each attribute $a \in A$ has a default value $\delta(a)$. We introduce the function $\alpha : V^* \times A \to \mathbb{N}$ to represent an assignment of attribute values to nodes.

$$\alpha(w, a) = \begin{cases} \alpha_C(w, a) & \text{for } w \in C\\ \delta(a) & \text{for } w = \mathsf{root}\\ @_a^{\tau(w)}(\alpha(\mathsf{mother}(w), a)) & \text{for } w \in E \end{cases}$$

where $@_a^t : \mathbb{N} \to \mathbb{N}$ is the function which computes the value of attribute a inside a tag t given its value outside. For most attributes it is the identity function.

There are two difficulties in the last equation: (1) computing α at the unknown point $(\mathsf{mother}(w), a)$ and (2) computing $@_a^{\tau(w)}$ at the unknown point $\alpha(\mathsf{mother}(w), a)$. This can be achieved using the following reduction:

$$\begin{aligned}
\alpha(w', a) &= \langle \alpha(w_1, a), \ldots, \alpha(w_n, a)\rangle[w']\\
@_a^t(x) &= \langle @_a^t(1), \ldots, @_a^t(k_a)\rangle[x]
\end{aligned}$$

where we assume that the possible values for attribute a range from 1 to k_a.

Furthermore, for all $w'' \in C$ it must hold that

$$\alpha(w'', a) = \alpha(\mathsf{mother}(w''), a)$$

We write S for the data range $\{1, \ldots, n_C\}$ and introduce the scope function $\sigma : V^* \to 2^S$. For all $w \in V^*$, its scope is an interval:

$$\mathsf{convex}(\sigma(w))$$

furthermore, it can be computed inductively:

$$\sigma(w) = \begin{cases} S & \text{if } w = \mathsf{root}\\ \{\pi(w)\} & \text{if } w \in C\\ \cup\{\sigma(w') \mid w' \in \mathsf{daughters}(w)\} & \text{if } w \in E \end{cases}$$

An element $w \in E$ is unused iff it is empty. Furthermore, in that case it is an immediate daughter of the root.

$$\begin{aligned} \tau(w) = \epsilon \qquad & \equiv \ |\sigma(w)| = 0 \\ \tau(w) = \epsilon \qquad & \Rightarrow \ \mathsf{root} = \mathsf{mother}(w) \end{aligned}$$

Finally, we introduce a cost function $\kappa : E \to \mathbb{N}$:

$$\kappa(w) = \langle \mathsf{cost}_T(t_1), \ldots, \mathsf{cost}_T(t_k) \rangle [\tau(w)]$$

where we pose $T^* = \{t_1, \ldots, t_k\}$ and $\mathsf{cost}_T(\epsilon) = 0$. The goal is to minimize the total cost:

$$\Sigma\{\kappa(w) \mid w \in E\}$$

In practice it is of course useful to break symmetries. For example the set $V = \{w_1, \ldots, w_n\}$ should be ordered using depth in the tree as primary sort key, minimal element as secondary sort key, and tag as tertiary sort key. It is easy to combine these keys together using an appropriate base, so that elements in V must have strictly increasing sort keys.

Breaking Symmetries.

Let's now consider some methods for breaking symmetries. Of these, we just implemented the first one.

Depth Method. Posing $E^* = E \uplus \{\mathsf{root}\}$, we introduce a function $\mathsf{depth} : E^* \to \mathbb{N}$ to represent the depth of an element in the tree representation:

$$\begin{aligned} \mathsf{depth}(\mathsf{root}) &= 0 \\ \mathsf{depth}(w) &= 1 + \mathsf{depth}(\mathsf{mother}(w)) \qquad \text{for } w \in E \end{aligned}$$

As usual the latter equation translates into:

$$\mathsf{depth}(w) = 1 + \langle \mathsf{depth}(\mathsf{root}), \mathsf{depth}(w_1), \ldots, \mathsf{depth}(w_{n_E}) \rangle [\mathsf{mother}(w)]$$

for the obvious appropriate encoding of E^*. Our first symmetry break is:

$$\mathsf{depth}(w_1) \leq \cdots \leq \mathsf{depth}(w_{n_E})$$

in other words, elements are ordered by increasing depth. Thus w_1 will always be a child of root, and earlier elements are used for higher levels in the tree and later elements for deeper levels.

Sequenced Union Method. A more clever method, is to say that the mother of an element must always precede it in the order $\mathsf{root} \prec w_1 \prec \cdots \prec w_{n_E}$:

$$\mathsf{mother}(w) \prec w \qquad \text{for } w \in V$$

and additionally if w is mother of w' and w'' and $w' \prec w''$ then all the elements in $\mathsf{down}(w')$ must precede all the elements in $\mathsf{down}(w'')$. For this, we introduce the functions $\mathsf{down}_E : E \to 2^E$, $\mathsf{eqdown}_E : E \to 2^E$, and $\mathsf{daughters}_E : E \to 2^E$ defined by:

$$\begin{aligned} \mathsf{down}_E(w) &= E \cap \mathsf{down}(w) \\ \mathsf{eqdown}_E(w) &= E \cap \mathsf{eqdown}(w) \\ \mathsf{daughters}_E(w) &= E \cap \mathsf{daughters}(w) \end{aligned}$$

and the following constraint must be satisfied:

$$\mathsf{down}_E(w) = \uplus\langle \mathsf{eqdown}_E(w_1), \ldots, \mathsf{eqdown}_E(w_{n_E})\rangle[\mathsf{daughters}_E(w)]$$

This is an application of the *sequenced union selection* constraint. It combines the *union selection* constraint with the sequentiality constraint.

To further break symmetries and strengthen propagation, we can require that $\mathsf{eqdown}_E(w)$ be an interval. In this fashion (1) the root of a subtree must be the min element of its eqdown_E and (2) we get stronger propagation since we know that there can be no hole in the eqdown_E:

$$\mathsf{convex}(\mathsf{eqdown}_E(w))$$

Finally, we should require that the unused elements occur at the end of the precedence order, i.e. for $w_i, w_{i+1} \in E$:

$$|\mathsf{down}(w_i)| = 0 \quad \Rightarrow \quad |\mathsf{down}(w_{i+1})| = 0$$

Note that scope should also be constrained using the sequenced union selection constraint:

$$\sigma(w) = \uplus\langle \sigma(w_1), \ldots, \sigma(w_n)\rangle[\mathsf{daughters}(w)]$$

i.e. not only is the scope of the mother formed from the union of the scopes of its daughters, but these scopes must also occur in sequence according to the precedence order $\prec$ of elements (i.e. the indices used to encode them). Note that, if we can identify $\pi(w)$ with w for $w \in C$, then we also have the following:

$$\mathsf{down}(w) = \mathsf{down}_E(w) \cup \sigma(w)$$

Stacking Method. This last method is concerned with symmetries of the following kind:

```
<B><I>...</I></B>
<I><B>...</B></I>
```

whenever w is mother of a single child w' then w must be assigned a *smaller* tag than w'. Thus, here we must assume a total order $\prec$ on tags, and the constraint is:

$$w = \mathsf{mother}(w') \ \wedge \ |\mathsf{daughters}(w)| = 1 \quad \Rightarrow \quad \tau(w) \preceq \tau(w')$$

For this to work, it must be the case that when switching the 2 tags is semantically significant then the configuration which does not satisfy the above constraint is redundant. I don't know if this is the case. Note that we used $\preceq$ rather than the stricter $\prec$ to allow for such stackings as `<U><U>...</U></U>`. For this method to work, root must be assigned some fictitious and lowest tag.