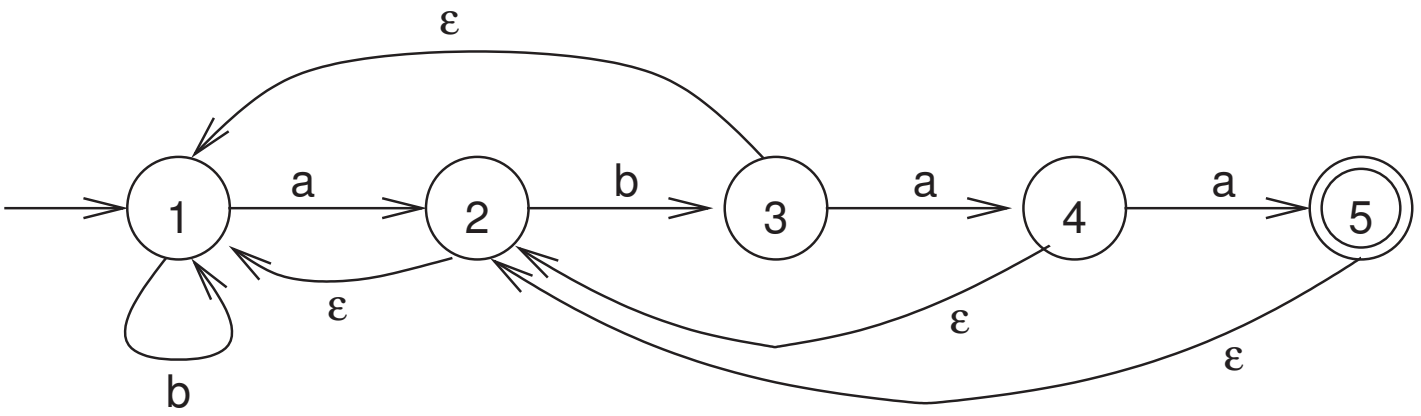
Factorisation in Algorithm Design

Roland Backhouse

RAMiCS, Groningen, October 2018

Knuth-Morris-Pratt Pattern Matching



"Factor graph" of Σ^*abaa (where $\Sigma = \{a,b\}$).

Reflexive-transitive *reduction* of Conway's "factor matrix" of Σ^*abaa.

Factor Matrix of $\Sigma^*aba a$

$$\begin{bmatrix}
\Sigma^* & \Sigma^*a & \Sigma^*ab & \Sigma^*aba & \Sigma^*abaa \\
\Sigma^* & \varepsilon+\Sigma^*a & (\varepsilon+\Sigma^*a)b & (\varepsilon+\Sigma^*a)ba & (\varepsilon+\Sigma^*a)baa \\
\Sigma^* & \Sigma^*a & \varepsilon+\Sigma^*ab & (\varepsilon+\Sigma^*ab)a & (\varepsilon+\Sigma^*ab)aa \\
\Sigma^* & \Sigma^*a & (\varepsilon+\Sigma^*a)b & \varepsilon+(\varepsilon+\Sigma^*a)ba & (\varepsilon+(\varepsilon+\Sigma^*a)ba)a \\
\Sigma^* & \varepsilon+\Sigma^*a & (\varepsilon+\Sigma^*a)b & \varepsilon+(\varepsilon+\Sigma^*a)ba & \varepsilon+((\varepsilon+\Sigma^*a)baa)
\end{bmatrix}$$

Reflexive-transitive *closure* of the factor graph of $\Sigma^*aba a$.

A Simple Example

Given a sequence of numbers, calculate the second largest.

A Simple Example

Given a sequence of numbers, calculate the second largest.

- there is no $\otimes$ such that, for all sequences x and y,

$$2\mathrm{ndLargest}(x{\cdot}y) \;=\; 2\mathrm{ndLargest}(x) \otimes 2\mathrm{ndLargest}(y) \quad .$$

A Simple Example

Given a sequence of numbers, calculate the second largest.

- there is no $\otimes$ such that, for all sequences x and y,

$$2\mathrm{ndLargest}(x{\cdot}y) \;=\; 2\mathrm{ndLargest}(x) \otimes 2\mathrm{ndLargest}(y) \quad .$$

Solution: Generalise to calculating the largest and the second largest.

$$\mathrm{TwoLargest}(x{\cdot}y) \;=\; \mathrm{TwoLargest}(x) \otimes \mathrm{TwoLargest}(y)$$

where

$$(m, n) \otimes (p, q) \;=\; (m{\uparrow}p \;,\; (m{\downarrow}p){\uparrow}n{\uparrow}q) \quad .$$

(First component is largest. "$\uparrow$" denotes maximum, and "$\downarrow$" denotes minimum.)

Overview

- Language Problems
- Fusion Theorem
- Example Generalisations
- Factor Theory
- Conclusions

Easy Language Problems

(L denotes the language generated by some context-free grammar.)

Is-empty

$$L = \phi \quad ?$$

Nullable

$$\varepsilon \in L \quad ?$$

Shortest word-length

$$\#L \quad .$$

Easy Language Problems

(L denotes the language generated by some context-free grammar.)

Is-empty

$$L = \phi \quad ?$$

Nullable

$$\varepsilon \in L \quad ?$$

Shortest word-length

$$\#L \quad .$$

Hard(er) Language Problems

Parsing: given x, determine

$$x \in L \quad ?$$

Inclusion: given M, determine

$$L \subseteq M \quad ?$$

Easy or Hard?

(L denotes the language generated by some context-free grammar.)

Do all words in L have even length?

$$L \subseteq (\Sigma\Sigma)^* \quad ?$$

Easy or Hard?

(L denotes the language generated by some context-free grammar.)

Do all words in L have even length?

$$L \subseteq (\Sigma\Sigma)^* \quad ?$$

Are all words in L also in the regular language E?

$$L \subseteq E \quad ?$$

Are all words in the regular language E also in L?

$$E \subseteq L \quad ?$$

Easy Language Problems

$$S \ ::= \ bSS \mid \varepsilon \ .$$

Is-empty

$$S = \phi \quad \equiv \quad (\{b\} = \phi \vee S = \phi \vee S = \phi) \ \wedge \ \{\varepsilon\} = \phi \ .$$

Nullable

$$\varepsilon \in S \quad \equiv \quad (\varepsilon \in \{b\} \wedge \varepsilon \in S \wedge \varepsilon \in S) \ \vee \ \varepsilon \in \{\varepsilon\} \ .$$

Shortest word length

$$\#S \ = \ (\#b + \#S + \#S) \downarrow \#\varepsilon \ .$$

Easy versus Hard

Consider, for example:

$$S \;::=\; bSS \mid \varepsilon \;\; .$$

Easy

Nullable

$$\varepsilon \in S \quad \equiv \quad (\varepsilon \in \{b\} \wedge \varepsilon \in S \wedge \varepsilon \in S) \;\vee\; \varepsilon \in \{\varepsilon\} \;\; .$$

Hard

In general,

$$x \in S \quad \not\equiv \quad (x \in \{b\} \wedge x \in S \wedge x \in S) \;\vee\; x \in \{\varepsilon\} \;\; .$$

For example,

$$bb \in S \quad \not\equiv \quad (bb \in \{b\} \wedge bb \in S \wedge bb \in S) \;\vee\; bb \in \{\varepsilon\} \;\; .$$

Fusion

Many problems are expressed in the form

$$evaluate \circ generate$$

where $generate$ generates a (possibly infinite) candidate set of solutions, and $evaluate$ selects a best solution.

Examples:

$$shortest \circ path \ ,$$

$$(x\in) \circ L \ .$$

Solution method is to *fuse* the generation and evaluation processes, eliminating the need to generate all candidate solutions.

Conditions for Fusion

Fusion is made possible when

- `evaluate` is an adjoint in a *Galois connection*,
- `generate` is expressed as a *fixed point*.

Solution method typically involves *generalising* the problem.

Goal of generalisation is to enable *factorisation*.

Galois Connections

Suppose $\mathcal{A} = (A, \sqsubseteq)$ and $\mathcal{B} = (B, \preceq)$ are partially ordered sets and suppose $F \in A \leftarrow B$ and $G \in B \leftarrow A$. Then (F, G) *is a Galois connection between* $\mathcal{A}$ *and* $\mathcal{B}$ iff, for all $x \in B$ and $y \in A$,

$$F(x) \sqsubseteq y \;\equiv\; x \preceq G(y) \quad .$$

Examples

Negation:

$$\neg p \Rightarrow q \;\equiv\; p \Leftarrow \neg q \quad .$$

Ceiling function:

$$\lceil x \rceil \leq n \;\equiv\; x \leq n \quad .$$

Maximum:

$$x \uparrow y \leq z \;\equiv\; x \leq z \wedge y \leq z \quad .$$

Even (divisible by two):

$$\mathsf{if}\ b \rightarrow 2 \;\square\; \neg b \rightarrow 1\ \mathsf{fi} \setminus m \;\equiv\; b \Rightarrow \mathsf{even}(m) \quad .$$

Parsing

$$x \in S \Rightarrow b \quad \equiv \quad S \subseteq \text{if } b \to \Sigma^* \,\square\, \neg b \to \Sigma^* - \{x\} \text{ fi} \quad .$$

Shortest Word

Let $\Sigma^{\geq k}$ denote the set of all words over alphabet Σ of length at least k.

Let $\#S$ denote the length of a shortest word in the language S.

$$\#S \geq k \quad \equiv \quad S \subseteq \Sigma^{\geq k} \quad .$$

Fusion Theorem

$$\mathsf{F}(\mu_{\preceq} g) \;=\; \mu_{\sqsubseteq} h$$

provided that

- F is a lower adjoint in a Galois connection of $\sqsubseteq$ and $\preceq$ (see brief summary of definition below)
- $\mathsf{F} \circ g \;=\; h \circ \mathsf{F}$.

Galois Connection

$$\mathsf{F}(x) \sqsubseteq y \;\equiv\; x \preceq \mathsf{G}(y) \quad .$$

F is called the *lower* adjoint and G the *upper* adjoint.

Language Recognition

Problem: For given word x and grammar G, determine $x \in L(G)$.
That is, implement

$$(x\in) \circ L \quad .$$

Language $L(G)$ is the least fixed point (with respect to the subset relation) of a monotonic function.

$(x\in)$ is the lower adjoint in a Galois connection of languages (ordered by the subset relation) and booleans (ordered by implication).
(Recall,

$$x\in S \Rightarrow b \quad \equiv \quad S \subseteq \text{if } b \to \Sigma^* \,\square\, \neg b \to \Sigma^* - \{x\} \text{ fi} \quad .)$$

Nullable Languages

Problem: For given grammar G, determine $\varepsilon \in L(G)$.

$$(\varepsilon \in) \circ L$$

Solution: Easily expressed as a fixed-point computation.

Works because:

- The function $(x\in)$ is a lower adjoint in a Galois connection (for all x, but in particular for $x = \varepsilon$).
- For all languages S and T,

$$\varepsilon \in S{\cdot}T \quad \equiv \quad \varepsilon \in S \wedge \varepsilon \in T \ .$$

General Parsing Problem

Problem: For given grammar G, and given word x, determine $x \in L(G)$.

$$(x \in) \circ L$$

Solution: Not directly expressible as a fixed-point computation.

Because:

- The function $(x\in)$ is a lower adjoint in a Galois connection.
- But, for $x \neq \varepsilon$, there is no operator $\otimes$ such that, for all S and T,

$$x \in S{\cdot}T \quad \equiv \quad (x \in S) \otimes (x \in T) \quad .$$

The Generalisation

Fusion *fails* because: for all x, $x \neq \varepsilon$, there is no $\otimes$ such that, for all words u and v,

$$x = uv \quad \equiv \quad (x = u) \otimes (x = v) \quad .$$

Solution: Suppose $x[i..j)$ is a segment of x (indexed by i and j). Then,

$$x[i..j) = uv \quad \equiv \quad \langle \exists k :: x[i..k) = u \ \wedge \ x[k..j) = v \rangle \quad .$$

Thus, if $R(w)$ denotes the relation $\langle i, j :: x[i..j) = w \rangle$,

and $B \bullet C$ denotes the composition of relations B and C,

$$R(uv) \quad = \quad R(u) \bullet R(v) \quad .$$

Cocke-Younger-Kasami

Let $F(L)$ denote the relation $\langle i, j :: x[i..j) \in L \rangle$.

$F(L)$ can be expressed as a fixed point: Interpret the grammar defining L in the algebra of relations, interpreting choice as set union, and product as composition of relations; interpret terminal symbol b as the relation on indices i and j given by $j = i+1 \wedge x[i..j) = b$.

Works because:

- The function F is a lower adjoint.
- For all languages S and T,

  $$F(S \cdot T) \quad = \quad F(S) \bullet F(T)$$

  where $B \bullet C$ denotes the composition of relations B and C.
- The $(0, N)$th entry of $F(L)$ is the required value. That is,

  $$(0, N) \in F(L) \quad \equiv \quad x \in L \quad .$$

Language Inclusion

Problem: For fixed (regular) language E and varying S, determine

$$\mathsf{S} \subseteq \mathsf{E} \quad .$$

Example: Emptiness test:

$$\mathsf{S} \subseteq \phi \quad .$$

Example: Pattern Matching: given pattern P, for each prefix t of text T, evaluate:

$$\{\mathsf{t}\} \subseteq \Sigma^* \cdot \{\mathsf{P}\} \quad .$$

Example: All words are of even length:

$$\mathsf{S} \subseteq (\Sigma \cdot \Sigma)^* \quad .$$

Language Inclusion

Problem: For fixed (regular) language E and varying context-free grammar G, determine

$$\mathsf{E} \subseteq \mathsf{L}(\mathsf{G}) \quad ?$$

Unsolvable!

Even for $\mathsf{E} = \Sigma^*$. (I.e. does G generate all words?)

Language Inclusion

Problem: For fixed (regular) language E and varying context-free grammar G, determine

$$L(G) \subseteq E \quad .$$

- Function $(\subseteq E)$ is a lower adjoint. Specifically,

$$S \subseteq E \Leftarrow b \quad \equiv \quad S \subseteq \text{if } b \rightarrow E \;\Box\; \neg b \rightarrow \Sigma^* \text{ fi} \quad .$$

- But, for $E \neq \phi$ and $E \neq \Sigma^*$, there is no $\otimes$ such that, for all languages S and T,

$$S \cdot T \subseteq E \quad \equiv \quad (S \subseteq E) \otimes (T \subseteq E) \quad .$$

Solution (Oege de Moor): Use factor theory to derive generalisation.

Factors

For all languages S, T and U,

$$S \cdot T \subseteq U \quad \equiv \quad T \subseteq S \backslash U \ ,$$

$$S \cdot T \subseteq U \quad \equiv \quad S \subseteq U / T \ .$$

Note:

$$S \backslash (U / T) \ = \ (S \backslash U) / T \ .$$

Hence, write

$$S \backslash U / T \ .$$

$S \backslash U$ is a *right factor* of U.

U / T is a *left factor* of U.

$S \backslash U / T$ is a *factor* of U.

The Factor Matrix

Let $\mathcal{L}$ denote the set of left factors of E.

Define the *factor matrix* of E to be the binary operator $\backslash$ restricted to $\mathcal{L}\times\mathcal{L}$. Thus entries in the matrix take the form $\mathsf{L}_0\backslash\mathsf{L}_1$ where L_0 and L_1 are left factors of E.

The factor matrix of E is denoted by $[\![\mathsf{E}]\!]$. It is a reflexive, transitive matrix.

$$[\![\mathsf{E}]\!] \quad = \quad [\![\mathsf{E}]\!]^* \quad .$$

The factor matrix of E is finite iff E is regular.

The factor matrix of a regular language E has a unique minimal starth root: the *factor graph* of E.

Using the Factor Matrix

Problem: For fixed regular language E and varying S, determine

$$\mathsf{S} \subseteq \mathsf{E} \quad .$$

Generalisation: For fixed regular language E and varying S, determine the relation

$$\mathsf{S} \subseteq [\![\mathsf{E}]\!] \quad .$$

(Formally, the relation $\langle \mathsf{L}, \mathsf{M} :: \ \mathsf{S} \subseteq \mathsf{L} \backslash \mathsf{M} \rangle$ where L and M range over the left factors of E.)

Works because:

$$\mathsf{S} \cdot \mathsf{T} \subseteq [\![\mathsf{E}]\!] \quad \equiv \quad (\mathsf{S} \subseteq [\![\mathsf{E}]\!]) \bullet (\mathsf{T} \subseteq [\![\mathsf{E}]\!]) \quad .$$

where $\mathsf{B} \bullet \mathsf{C}$ denotes the composition of relations B and C.

All Even

Problem: For given grammar G, determine whether all words in $\mathsf{L}(\mathsf{G})$ have even length. I.e. implement

$$\mathsf{alleven} \circ \mathsf{L} \quad .$$

The function alleven is a lower adjoint in a Galois connection. Specifically, for all languages S,

$$\mathsf{alleven}(\mathsf{S}) \Leftarrow \mathsf{b} \quad \equiv \quad \mathsf{S} \subseteq \mathsf{if}\ \neg\mathsf{b} \rightarrow \Sigma^* \ \square\ \mathsf{b} \rightarrow (\Sigma{\cdot}\Sigma)^*\ \mathsf{fi} \quad .$$

Nevertheless, fusion *doesn't* work (directly) because

- there is no $\otimes$ such that, for all languages S and T,

$$\mathsf{alleven}(\mathsf{S}{\cdot}\mathsf{T}) \quad \equiv \quad \mathsf{alleven}(\mathsf{S}) \otimes \mathsf{alleven}(\mathsf{T}) \quad .$$

Solution: Generalise by tupling: compute simultaneously alleven and allodd.

Solution is predicted by factor theory: the factors of $(\Sigma{\cdot}\Sigma)^*$ are

$$(\Sigma{\cdot}\Sigma)^* \quad , \quad \Sigma \cdot (\Sigma{\cdot}\Sigma)^* \quad , \quad \Sigma^* \quad , \quad \phi \quad .$$

The last two are necessary because some nonterminals may be "useless". The informal solution is strictly incorrect.

Conclusion

- A good theory helps to break poblems down.

  regular algebra

  Galois connections

  fixed-point calculus

- Factorisation is the key to efficiency.
- Know your algebra!