

Programmbibliotheken für die Verarbeitung Endlicher Automaten

Armin Schmidt <armin.sch@gmail.com>

HS Endliche Automaten für die Sprachverarbeitung (Karin Haenelt)
SS 08, Seminar für Computerlinguistik, Universität Heidelberg

14. Juli 2008

- 1 Auswahlkriterien
- 2 Bibliotheken
- 3 Experimente
- 4 Referenzen

Auswahlkriterien für EA-Bibliotheken

- DFSA, FST, WFST?
- benötigte Operationen
- Textverarbeitung: Unicode-Kompatibilität
- reguläre Ausdrücke
- Größe/Format der kompilierten Automaten (*lazy Impl.*)
- Laufzeit
- Programmiersprache
- externe Abhängigkeiten
- Plattformunabhängigkeit
- Stabilität
- Dokumentation
- Lizenz
- andere Schnittstellen: Kommandozeilentool, GUI-Werkzeug

Vorgestellte Bibliotheken

- Bibliotheken, keine reinen Kommandozeilen- o.a. Werkzeuge
- möglichst mit Kommandozeilentools
- Sprache: C/C++
- zur Sprachverarbeitung (nicht exklusiv)
- stabil, aktuell, gewartet
- dokumentiert
- (mehr oder weniger) freie Lizenz



RWTH FSA Toolkit (Kanthak und Ney, 2004)

<http://www-i6.informatik.rwth-aachen.de/~kanthak/fsa.html>

Automatenfamilien	WFST
Programmiersprache	C++
ext. Abhängigkeiten	keine
Ausgabe-Format	AT&T, binär, XML, dot
Version	0.9.4
Dokumentation	ausführliche Readme, Mailing-List-Archive
Lizenz	Qt Public License / Qt Non-Commercial License - Derivat
zuletzt aktualisiert	2005
andere Schnittstellen	Kommandozeile, Python

Operationen (Auswahl)

Grundlegende	<i>sort, cache</i>
Rationale	<i>reversal, project-input, project-output, union, concat, invert</i>
Graph	<i>depth-first-search, single-source shortest path strongly-connected components</i>
Mengenrelationen	<i>compose, intersect, complement</i>
Äquivalenztransf.	<i>determinize, minimize, remove-ϵ</i>
Suchalgorithmen	<i>best, n-best</i>
Gewichte	<i>prune, posterior, push, failure</i>
Diagnose	<i>count</i>

AT&T FSM Library (Mohri et. al, 1998)

<http://www.research.att.com/~fsmtools/fsm/>

Automatenfamilien	WFST
Programmiersprache	C
ext. Abhängigkeiten	keine
Ausgabe-Format	AT&T-FSM-Format, dot
Version	4.0
Dokumentation	MAN-Pages
Lizenz	Non-Commercial kein Open Source
zuletzt aktualisiert	2004
andere Schnittstellen	Kommandozeile

Operationen (Auswahl)

Grundlegende	<i>sort</i>
Rationale	<i>union, concat, Kleene closure, reversal invert, project</i>
Graph	–
Mengenrelationen	<i>compose, intersect, difference</i>
Äquivalenztransf.	<i>remove-ϵ, determinize, minimize, remove inaccessible states/transitions</i>
Suchalgorithmen	<i>best, n-best</i>
Gewichte	<i>prune</i>
Diagnose	<i>graphical, access/mutate states/transitions</i>

AT&T GRM Library (Mohri, 2001)

- <http://www.research.att.com/~fsmtools/grm/>
- wie AT&T FSM Library
- Erweiterungen zur Konstruktion, Kompilation und Modifikation von Grammatiken

Erweiterungen (Auswahl)

- Kontext-abhängige Regeln → WFST
- Konstruktion gewichteter Suffix-Automaten
- Counting-Sequences in WFSAs
- Statistische Sprachmodelle

C++-Templates

- erlauben die Definition von Klassen und Funktionen mit generischen Typen
 - ⇒ eine Klasse/Funktion für unbestimmte Datentypen
- Standard Template Library (STL)
 - definiert große Menge an Datentypen, Iteratoren, Algorithmen
 - ASTL als Erweiterung der STL

Beispiel

```
#include <iostream>
#include <string>
using namespace std;
template <class T>
T maximum(T x, T y) {
    if (y >= x)
        return y;
    else
        return x;
}
int main(void) {
    //Calling template function
    cout << maximum<int>(3,7) << endl; // 7
    cout << maximum<string>("abc", "xyz") << endl; // xyz
    return 0;
}
```

ASTL (Le Maout, 1998)

<http://astl.sourceforge.net>

Automatenfamilien	FSA
Programmiersprache	C++
ext. Abhängigkeiten	keine
Ausgabe-Format	FSA-Objekte
Version	beta2.0
Dokumentation	Referenz Dok. (unvollständig, zerstreut)
Lizenz	GPL
zuletzt aktualisiert	2003
andere Schnittstellen	keine

Operationen (Auswahl)

Grundlegende	<i>construct, compare, copy</i>
Rationale	???
Mengenrelationen	<i>compose, intersect, difference</i>
Äquivalenztransf.	<i>determinize, minimize</i>
Suchalgorithmen	???

Beispiel

```
#include <astl.h>
#include <dfa.h>
#include <vector>
#include <iterator>

int main() {
    using namespace std;
    DFA_matrix<> A;
    DFA_matrix<>::state_type q = A.new_state();
    DFA_matrix<>::state_type p = A.new_state();
    A.set_trans(q, 'a', p);
    A.initial(q);
    A.final(p) = true;
}
```

WFST (Adant, 2000)

<http://membres.lycos.fr/adant/tfe/>

Automatenfamilien	WFST
Programmiersprache	C++
ext. Abhängigkeiten	ASTL, AT&T FSM, Perl 5.0
Ausgabe-Format	FSA-Objekte, AT&T FSM, dot
Version	???
Dokumentation	Referenz Dok.
Lizenz	GPL
zuletzt aktualisiert	2000?
andere Schnittstellen	keine

Operationen (Auswahl)

Rationale	<i>union, invert, compose, reversal, project</i>
Mengenrelationen	<i>compose, intersect, difference, complete, complement</i>
Äquivalenztransf.	<i>determinize, minimize, connect, remove-ϵ</i>
Suchalgorithmen	<i>best</i>



andere interessante Bibliotheken

- allgemein:
 - Grail (<http://www.csd.uwo.ca/Research/grail/>)
 - Carmel (<http://www.isi.edu/licensed-sw/carmel/>)
- reguläre Ausdrücke:
 - PCRE (<http://www.pcre.org>)
 - BOOST.REGEX
(http://www.boost.org/doc/libs/1_35_0/libs/regex/)
- Morphologie-Bau:
 - ALE-RA (<http://nl.ijs.si/et/Thesis/ALE-RA/>)
- Verarbeitung von Konfigurationsdateien
 - Augeas (<http://www.augeas.net>)

Setup (aus: Kanthak und Ney, 2004)

- PC mit
 - 1.2 GHz Athlon CPU, 2 GB Hauptspeicher, Linux
- Automat:
 - Verbmobil: Speech-Recognition-Network, voll-expandiert in FSA
 - 12.203.420 Zustände, 37.174.684 Übergänge

Setup (aus: Kanthak und Ney, 2004)

- PC mit
 - 1.2 GHz Athlon CPU, 2 GB Hauptspeicher, Linux
- Automat:
 - Verbmobil: Speech-Recognition-Network, voll-expandiert in FSA
 - 12.203.420 Zustände, 37.174.684 Übergänge

Ergebnisse

RWTH FSA	AT&T FSM	WFST
Hauptspeicherverbrauch in MB		
360	1500	> 1850*
CPU-Zeit in Sekunden		
105	515	> 40*

* abgebrochen wg. überschrittenen Speichergrenzen

- Arnaud Ardant, 2000, *WFST: A Finite-State Template Library in C++*, <http://membres.lycos.fr/adant/tfe/>
- S. Kanthak and H. Ney, 2004, *FSA: An Efficient and Flexible C++ Toolkit for Finite State Automata Using On-Demand Computation*, Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain, pp. 510-517, July, 2004
- Vincent Le Maout, 1998, *ASTL: Automaton Standard Template Library*, <http://astl.sourceforge.net>
- Mehryar Mohri, Fernando C. N. Pereira, and Michael Riley, 1998, *A Rational Design for a Weighted Finite-State Transducer Library*, Lecture Notes in Computer Science, 1436, 1998
- Mehryar Mohri, 2001, *Weighted Grammar Tools: the GRM Library*. In Jean claude Junqua and Gertjan van Noord, editors, *Robustness in Language and Speech Technology*. pages 165-186. Kluwer Academic Publishers, The Netherlands, 2001