Compiling Context Restrictions with Nondeterministic Automata (that Resemble Bimachines)

Anssi Yli-Jyrä (online slides with revisions on Sep 26, 2011)

University of Helsinki, Department of Modern Language
(The Academy of Finland Project #128536 *Open and Language Independent Automata-Based Resource Production Methods for Common Language Research Infrastructure*)

July 12, 2011
FSMNLP 2011, Blois, France

Background: Constraints

Many NLP tasks generate tentative output strings for an input string:

- a 'word lattice' in acoustic speech recognization
- a lexical transducer in concatenative morphotactics
- input-output pairs in phonological model
- ambiguous words in morphological disambiguation etc.

The outputs $w \in O \subseteq \Sigma^*$ may be reduced by choosing a language model $P : \Sigma^* \to [0, 1]$ and by maximizing the probability of the chosen element(s) when

$$O' = \arg\max_{w \in O} P(w)$$

If the image of P is $\{0, 1\}$, P characterizes a string set $L_P \subseteq \Sigma^*$ such that

$$O' = O \cap L_P$$

The string set L_P is called a constraint language. We are now interested in its compilation.

Background

Roche (1992) disambiguates the set of candidate analyses by removing the occurrences of *negative patterns* that constitute a regular language $\alpha \subseteq \Sigma^*$ (confusingly aka a *local grammar*).

Definition

The set of negative patterns α is used to define a string property:

$$\text{forall } 0 \leq i \leq j \leq n.[c_{i+1}...c_j \in \alpha \rightarrow \text{false}].$$

i.e.

$$\text{forall } 0 \leq i \leq j \leq n.[c_{i+1}...c_j \notin \alpha].$$

The property defines a regular constraint language

$$L_P = \overline{\Sigma^* \alpha \Sigma^*}.$$

We are now going to look at generalizations of this definition.

Background

Context restrictions (CRs) (Koskenniemi 1983) are a non-trivial generalization of the negative patterns:

Definition

- A *simple context restriction* $\alpha \Rightarrow \lambda\_\rho$ contains a negative pattern language α and an excepted context*:

$$\text{forall } 0 \leq i \leq j \leq n. \left[c_{i+1} \ldots c_j \in \alpha \implies c_1 \ldots c_i \in \lambda \wedge c_{j+1} \ldots c_n \in \rho \right].$$

- A *multi-context context restriction* $\alpha \Rightarrow \lambda_1\_\rho_1, \ldots, \lambda_k\_\rho_k$ is similar, but with several excepted contexts:

$$\text{forall } 0 \leq i \leq j \leq n. \left[c_{i+1} \ldots c_j \in \alpha \implies \bigvee_{l=1}^{k} c_1 \ldots c_i \in \lambda_l \wedge c_{j+1} \ldots c_n \in \rho_l \right].$$

* Notational Convention (Deviates from Koskenniemi 1983)

Denote a pair of *total* contexts $\#\lambda\_\rho\#$ by λ_ρ.

Compilation of Simple Context Restriction Constraints

A Folklore Method ("SF", uses only classical star-free regular operations)

Let $\alpha \Rightarrow \lambda \_ \rho$ (with regular $\alpha, \lambda, \rho \subseteq \Sigma^*$) be a simple context restriction. It is then equivalent to

$$\overline{\overline{\lambda}\alpha\Sigma^* \cup \Sigma^*\alpha\overline{\rho}}.$$

A Generalization ("SF")

Let $\alpha \Rightarrow \lambda_1 \_ \rho_1, \lambda_2 \_ \rho_2$ be a two-context context restriction. It is then equivalent to

$$\overline{\overline{\lambda_2}\alpha\overline{\rho_1} \cap \overline{\lambda_1}\alpha\overline{\rho_2} \cap \overline{(\overline{\lambda_1} \cap \overline{\lambda_2})\alpha\Sigma^*} \cap \overline{\Sigma^*\alpha(\overline{\rho_1} \cap \overline{\rho_2})}}.$$

The expression for any larger number k of contexts is similar but grows exponentially in the number of contexts (Yli-Jyrä 2003).

The "SF" method is fast for small k but becomes impractical for less typical but still realistic context restrictions.

Context Restriction Constraints: Sample Use

Example

In the English Finite State Intersection Grammar, there is a rule

```
@>N    =>     DET .   _   ,
                      _   @    @lparen ,
 Too..>A @ . Adj  .   _   @  . <Indef> ,
             Adj  .   _   @  . @A<
                          @  . <Indef> ,
[quite | rather] .   _   @  . CENTRAL ,
                      _ [@   @comma . ]*
                          @  . [if | though | including] ,
         Nominal  .   _ [@  . [@A< | @>A | CC | NEG-PART |
                              @ADVL | PUNCT | @>N | de] . ]*
                          @  . Nominal . [ NPhead | @N< ] ,
     [GEN | NUM]  .   _   @  . POST
```

(operators: =>, _, [,], ,, |, *)
(language constants: ., Too..>A, Adj, Nominal, NPhead)

Generalized Restriction

A much more efficent compilation method is given by a two-marker approach (Yli-Jyrä & Koskenniemi 2004, reimplemented in Hulden 2008):

Theorem (Compiling through "GR" or "FO" approaches)

Let $\alpha \Rightarrow \lambda_1 \_ \rho_1, ..., \lambda_k \_ \rho_k$ be a multi-context context restriction. Then it is equivalent to

$$\Sigma^* - h_2\left(\Sigma^* \diamond \alpha \diamond \Sigma^* - \bigcup_{i=1}^{k} \lambda_i \diamond \Sigma^* \diamond \rho_i\right)$$

where $\diamond \cap \Sigma = \varnothing$ *and* $h_2(v \underbrace{\diamond x \diamond y}_{\text{contains 2 times } \diamond}) = vxy$.

The compilation method is encapsulated by the *generalized restriction* operator:

Definition (Generalized Restriction ("GR"))

Let W and W' be regular subsets of $\Sigma^* \diamond \Sigma^* \diamond \Sigma^*$ with disjoint Σ^* and $\diamond$. Then

$$W \overset{d\{\diamond\}}{\Rightarrow} W' = \Sigma^* - h_d(W - W') \text{ where } d \text{ is an integer.}$$

Further Algorithmic Challenges

It is well known that certain problems on regular expressions are PSPACE complete. GR does help in this, but it is a useful means to express regular languages in different ways.

Problem

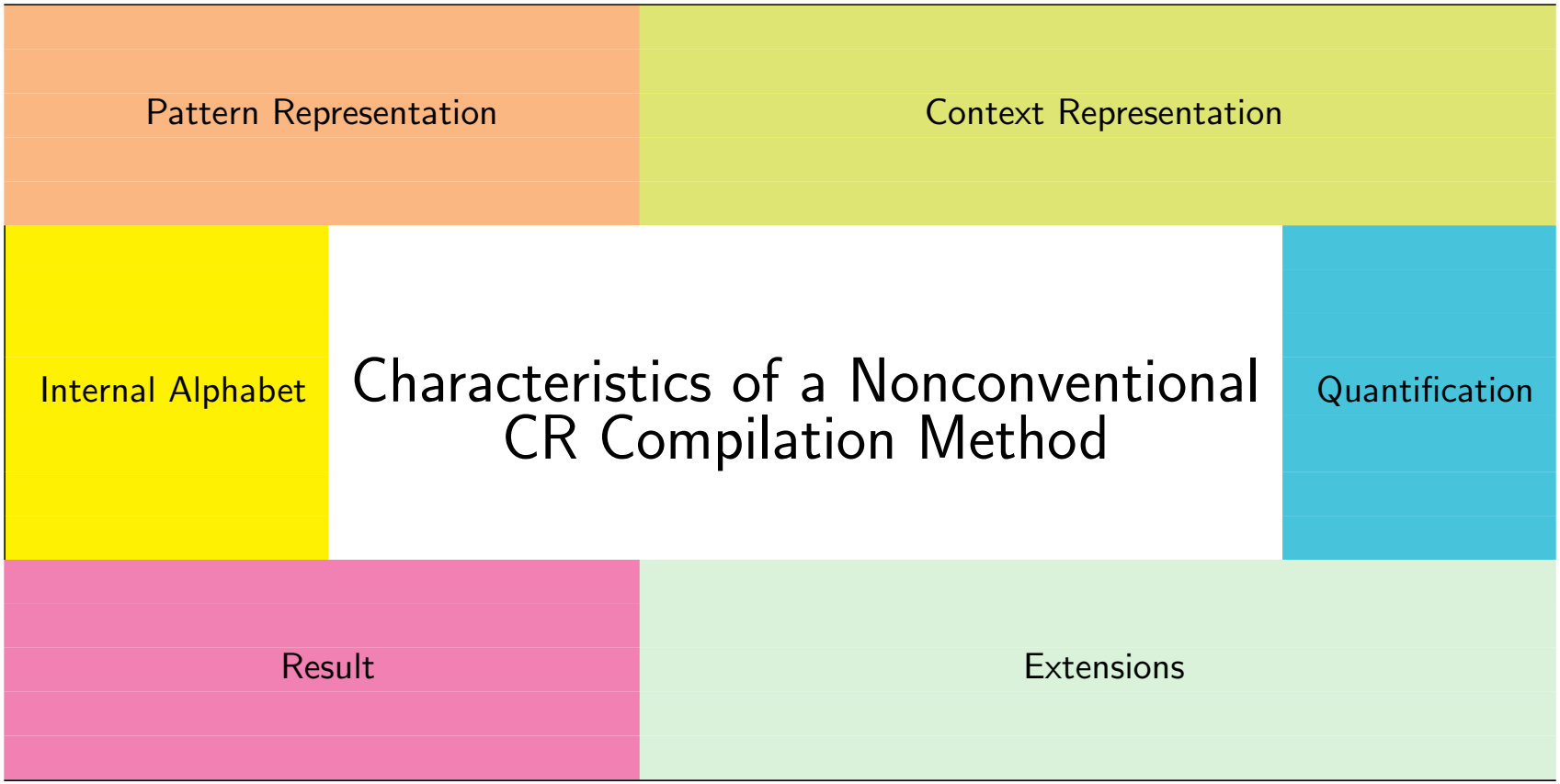
The GR method seems to be difficult to beat in many applications where context restrictions are needed. It is already very simple and dividing its arguments into pieces is seldom useful.

We still have situations where GR is not sufficiently efficient or applicable and we have endless hunger for optimizations.

- very large number of contexts / rule (e.g. 1M in contexted morpheme lexicon)
- large number of constraints / grammar (e.g. 2k in the English FSIG)
- long and patological contexts that blow-up under determinization
- constraints with weights

The New Approach

The **aim of this paper** is to propose an alternative compilation approach that wins GR in elegancy and even in its practical value, if possible. This alternative approach consists of the following interesting aspects:



Pattern Representation

Goal

Use a *patternless* GR

$$\left(\overline{\varnothing}\diamond\overline{\varnothing} \overset{1\{\diamond\}}{\Longrightarrow} W'\right) = \{c_1 \ldots c_n \mid \forall i \in \{0, \ldots, n\}. c_1 \ldots c_i \diamond c_{i+1} \ldots c_n \in W'\}$$

where $h_1(x \diamond y) = xy$ for all $x, y \in \overline{\varnothing}$ to compile an arbitrary CR $\alpha \Rightarrow \lambda_1 \_ \rho_1, \ldots, \lambda_k \_ \rho_k$.

Solution

$$[\alpha \Rightarrow \lambda_1 \_ \rho_1, \ldots, \lambda_k \_ \rho_k] = [\Sigma^* \diamond \alpha \diamond \Sigma^* \overset{2\{\diamond\}}{\Rightarrow} \cup_{i=1}^{k} \lambda_i \diamond \Sigma^* \diamond \rho_i] \quad (1)$$

$$[W \overset{2\{\diamond\}}{\Rightarrow} W'] = [\Sigma^* \diamond \Sigma^* \diamond \Sigma^* \overset{2\{\diamond\}}{\Rightarrow} W' \cup W''] \text{ where } [W'' = (\Sigma^* \diamond \Sigma^* \diamond \Sigma^* - W)] \quad (2)$$

$$[\Sigma^* \diamond \Sigma^* \diamond \Sigma^* \overset{2\{\diamond\}}{\Rightarrow} W''] = [\Sigma^* \diamond_1 \Sigma^* \diamond_2 \Sigma^* \overset{2\{\diamond_1, \diamond_2\}}{\Rightarrow} W''']$$
$$\text{with } v \diamond x \diamond y \in W' \cup W'' \Leftrightarrow v \diamond_1 x \diamond_2 y \in W''' \quad (3)$$

$$[\Sigma^* \diamond_1 \Sigma^* \diamond_2 \Sigma^* \overset{2\{\diamond_1, \diamond_2\}}{\Rightarrow} W'''] = \left(\overline{\varnothing}\diamond_1\overline{\varnothing} \overset{1\{\diamond_1\}}{\Longrightarrow} \left(\overline{\varnothing}\diamond_2\overline{\varnothing} \overset{1\{\diamond_2\}}{\Longrightarrow} W'''\right)\right) \quad (4)$$

Internal Alphabet

The table shows the internal alphabets used by some compilation methods mentioned in the paper:

	SF	IC	GR	IF	WL	Current
Σ	x					
$\Sigma \times \{1, ..., k\} \times \{<, >\}$		x				
$\Sigma \times \{1, ..., k\}$		x				
$\Sigma \cup \{\diamond\}$			x	x		
$\Sigma \times \mathbb{B} \times \mathbb{B}$					x	
$\Sigma \cup \{\diamond_1, \diamond_2\}$						x

Thus, using patternless GRs leads to a method whose alphabet is close to optimal (Σ). The internal alphabet resembles the alphabets of the GR and IF approaches but is not strictly the same.

Context Representation

To construct a recognizer for the patternless GR $\left(\overline{\varnothing}\diamond\overline{\varnothing} \overset{1\{\diamond\}}{\Longrightarrow} W'\right)$ where $W' \subseteq \overline{\varnothing}\diamond\overline{\varnothing}$, one normally computes first the relative complement $\overline{\varnothing}\diamond\overline{\varnothing} - W'$.

Goal

Compute the relative complement $\overline{\varnothing}\diamond\overline{\varnothing} - W'$ without determinizing the recognizer of W'.

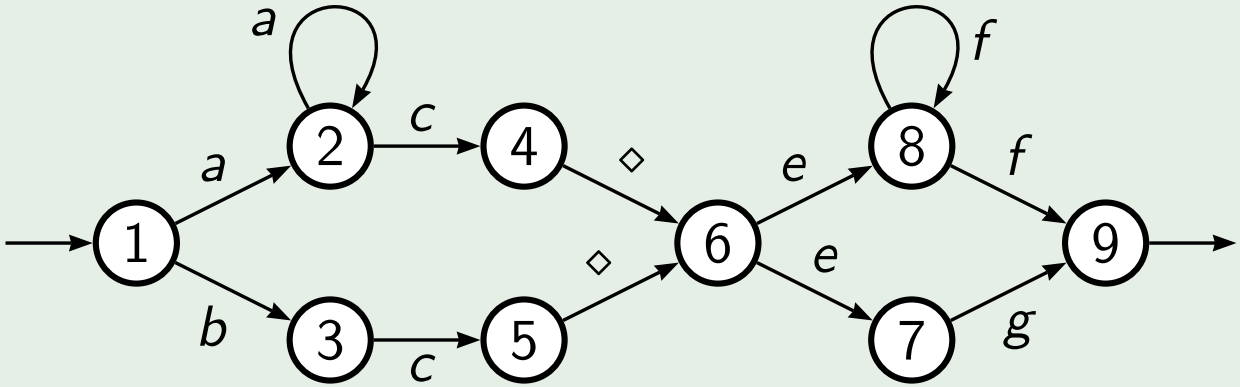
Solution

Do not determinize towards the final states. Instead, choose to determinize the recognizer of W' towards the $\diamond$-transitions.

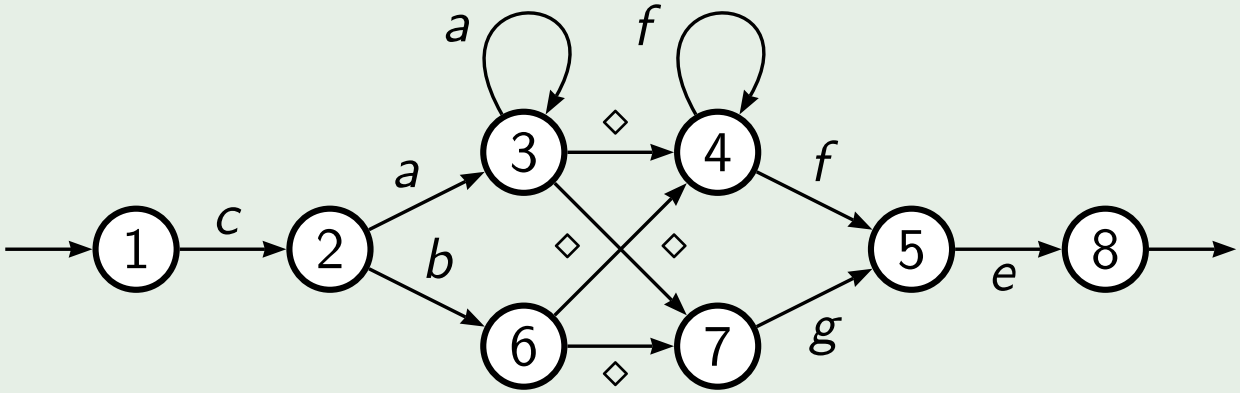
The result is called *inward deterministic* w.r.t. the marker $\diamond$.

Context Representation via Inward Deterministic NFAs

Example $(a^+c\_ef^+, a^+c\_eg, bc\_ef^+, bc\_eg)$



Example $(ca^+\_f^+e, ca^+\_ge, cb\_ge, cb\_f^+e)$



Quantification

Recall that every patternless GR expresses essentially a universal quantification:

$$\left(\overline{\varnothing}\diamond\overline{\varnothing} \overset{1\{\diamond\}}{\Longrightarrow} W'\right) = \{c_1 \ldots c_n \mid \forall i \in \{0, \ldots, n\}. c_1 \ldots c_i \diamond c_{i+1} \ldots c_n \in W'\}$$

Usually, one would compile this (like GR in general) with two complementations: $\left(\overline{\varnothing}\diamond\overline{\varnothing} \overset{1\{\diamond\}}{\Longrightarrow} W'\right) = \Sigma^* - h_1(\overline{\varnothing}\diamond\overline{\varnothing} - W')$ where $h_1(v \diamond y) = vy$ for all $v, y \in \Sigma^*$.

Goal

Avoid complementations when computing the universal quantification.

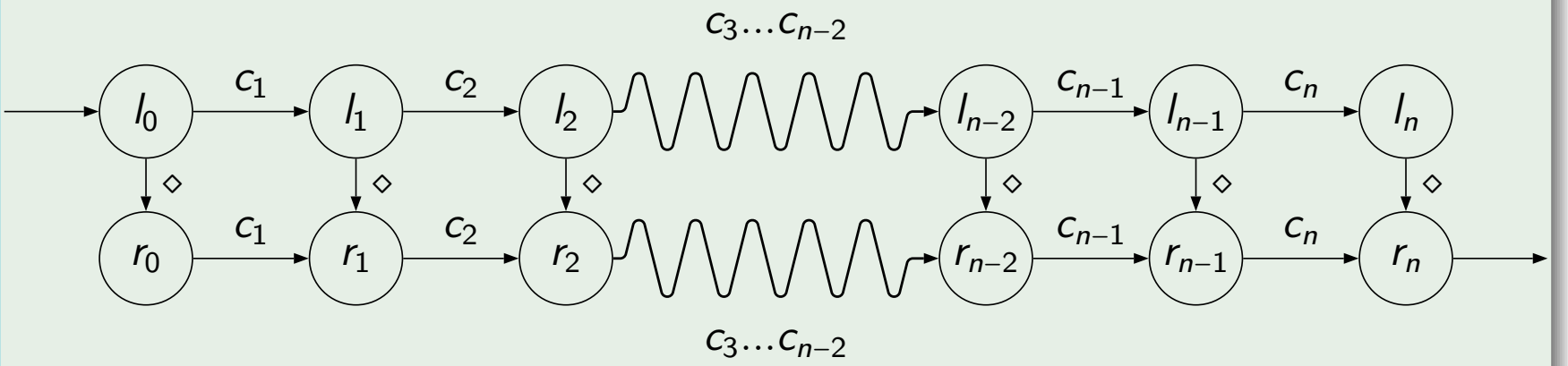
Solution

For each string $w \in \Sigma^*$ in the resulting language R, check that $h_1^{-1}(w) \subseteq W'$. That is, $h_1^{-1}(R) \cap W'$ is closed under variant markings.

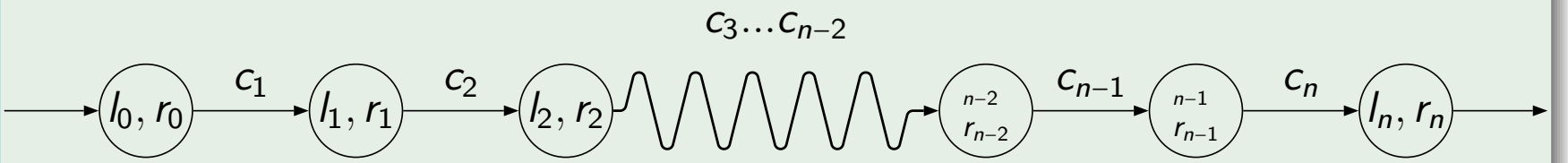
Quantification: Closure under Variant Markings

Example (A Ladder-Shaped Substructure in the Recognizer of W')

The paths on strings $h_1{-}1(c_1 \ldots c_n)$ in $\mathcal{A}$:



The string $h(h^{-1}(c_1 \ldots c_n)) = c_1 \ldots c_n$ corresponds to a path in a specially restricted product automaton:



Construction of the Result

Definition

Let $\mathcal{A}$ be an inward deterministic automaton with $\mathcal{L}(\mathcal{A}) \subseteq \Sigma^* \diamond \Sigma^*$. Then define

$$\textsc{Ladder}(\|\mathcal{A}\|) = \Sigma^* \diamond \Sigma^* - h_1^{-1}(h_1(\Sigma^* \diamond \Sigma^* - \|\mathcal{A}\|)).$$

Algorithm

Require: Inward det. NFA $\mathcal{A} = (Q, \{i\}, \{f\}, \delta)$ with $\mathcal{L}(\mathcal{A}) \subseteq \Sigma^* \diamond \Sigma^*$

1: $Q' \leftarrow \{(l, r) \mid (l, \diamond, r) \in \delta\}$;
$I' \leftarrow \{(l, r) \in Q' \mid l \in I\}$;
$F' \leftarrow \{(l, r) \in Q' \mid r \in F\}$

2: $\delta' \leftarrow \{((l, r), a, (l', r')) \mid (l, r), (l', r') \in Q', l' \in la, r' \in ra\}$

3: $\mathcal{A}' = (Q', I', F', \Sigma, \delta')$;

Ensure: $\|\mathcal{A}''\| = h_1(\textsc{Ladder}(\|\mathcal{A}\|))$

The result $\mathcal{A}$ is not deterministic, but its is unambiguous (at most one accepting path for each input).

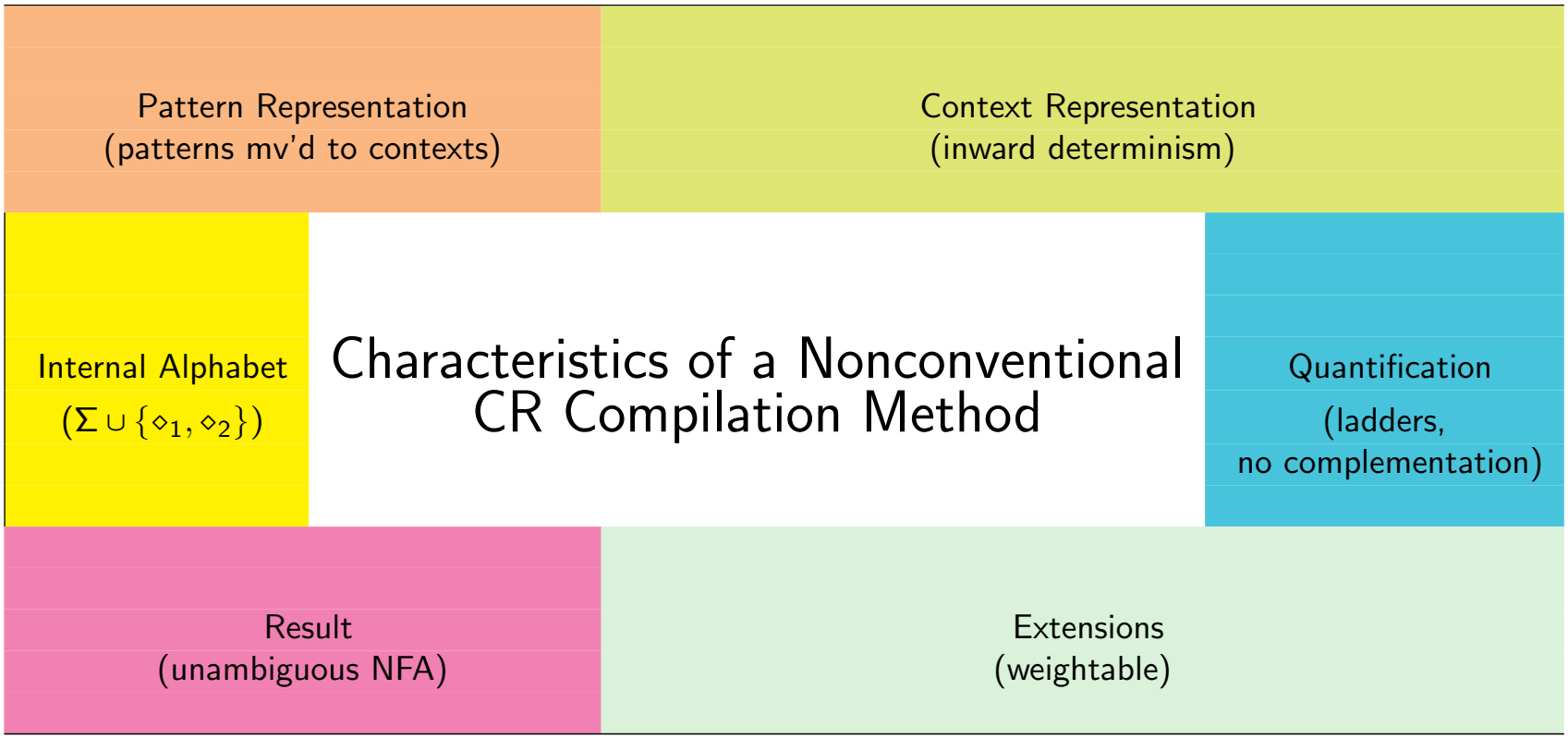
Extensions / Optimizations

The current method

- extends naturally with weighted contexts
  (manuscript 2011 / Acta Cybernetica)
- is generalizable with nested exceptions and more marked positions
- is optimizable with
  - failure transitions
  - minimization of inward deterministic automata
  - filtering of the recognizer for W' during inward determinization.

The Summary

We presensented a *new* compilation method for Koskenniemi's multi-context context restrictions (CRs), by reducing them to nested 1-marker GRs that are compiled directly, without language complementations.



Applications

CRs have many applications:

- constraints in Two Level Morphology
- temporal relations
- conditional rewriting rules
- context-dependent concatenation
- word sense disambiguation, finite-state intersection grammar
- ...

The current work may contain useful ideas for more widely applicable compilation algorithms.

Thanks for Your Interest!

Special thanks to Andreas Maletti, Jacques Sakarovitch and Måns Hulden for their clarifying questions.
(In the following, I reply to one comment I anticipated before the talk.)

Is the Current Method Based on Bimachines?

A Probable Comment

"The construction is just/is not an application of a composition of left and right sequential transducers."

My Reply

Finding ladder-shaped structures in an inward deterministic automaton bears *only very limited resemblance* to the well known behavior of bimachines:

- combining left and right sequential automata
- the product states.

The resemblance is less than equivalence, though. The main *crucial difference* is that the current construction maps, for each unmarked string $c_1 \ldots c_n$, exactly $n+1$ paths on strings $h_1^{-1}(c_1 \ldots c_n)$ to a path on an unmarked string, whereas a bimachine combines one left- and one right-deterministic path to one.