

Declarative
Syntactic Processing of Natural Language
Using Concurrent Constraint Programming
and Probabilistic Dependency Modeling

Irene Langkilde-Geary
Natural Language Technology Group
University of Brighton

The Problem

- ✦ Improve the quality of MT output using syntactic modeling
- ✦ Sampling of efforts
 - ✦ Charniak et al., 2003
 - ✦ Och et al., 2004
 - ✦ Galley et al., 2006
 - ✦ Has-san et al., 2007
 - ✦ Chang and Toutanova, 2006
 - ✦ Koehn and Hoang, 2007

Challenges

- ✦ Annotation for training
- ✦ Compatibility of representation across components
- ✦ Interdependence of realization subtasks, ie.,
 - ✦ lexical choice,
 - ✦ choice of syntactic structure,
 - ✦ word order,
 - ✦ etc.
- ✦ Combinatorial complexity and consequent computational (in)tractability

The Need for Declarativeness

- ★ Suggested by an accumulation of work related to generation system architectures

- ★ Cahill et al., 2000

- ★ Calder et al., 1999

- ★ Beale et al., 1998

- ★ Beale, 1997

- ★ Elhadad et al., 1997

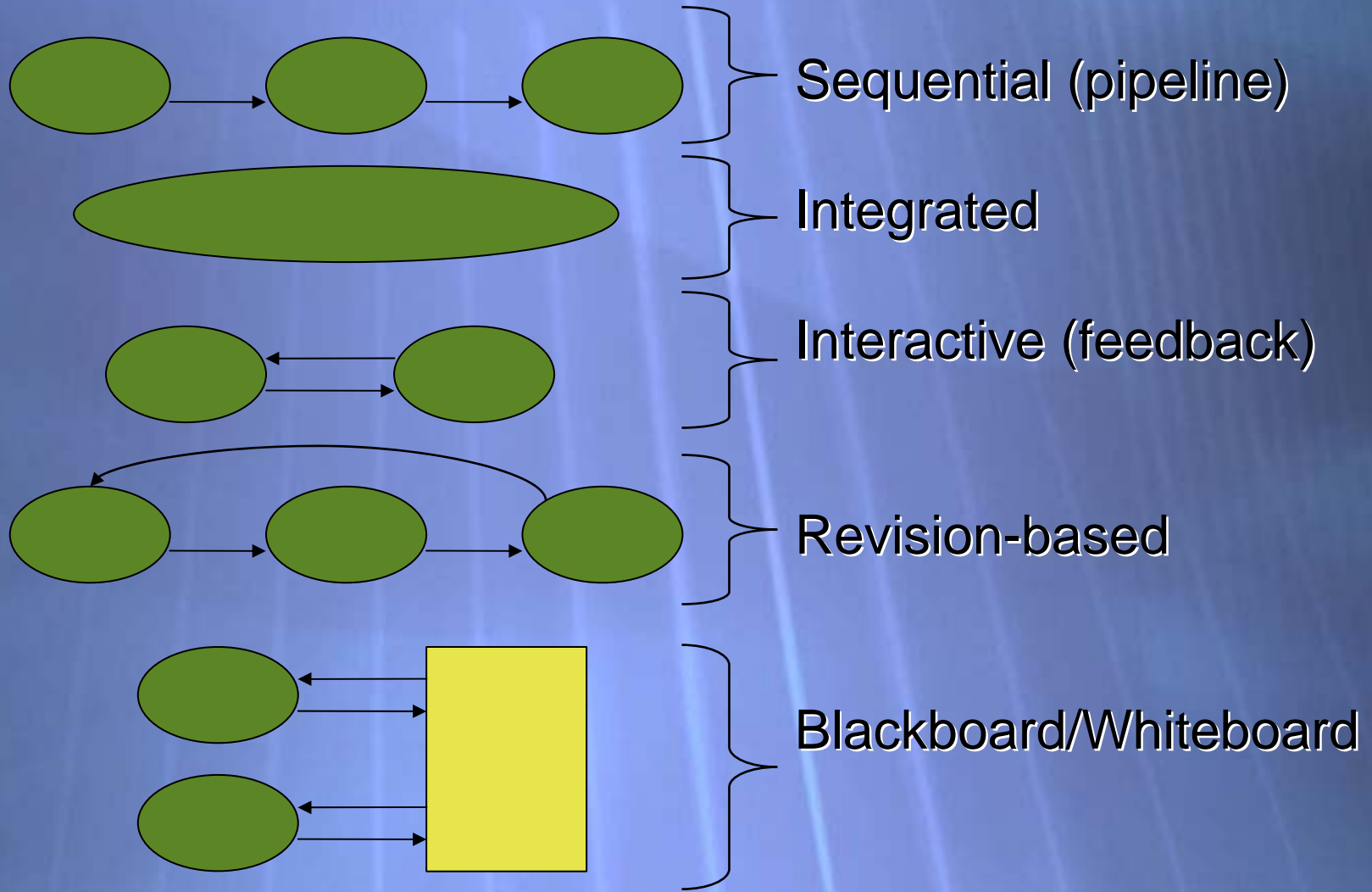
- ★ Smedt et al., 1996

- ★ Robin, 1994

- ★ Meteer, 1990

- ★ as well as own experience trying to integrate syntactic modeling into HALogen

Generation System Architectures



What does “declarative” mean?

- ✦ Representations are
 - ✦ Stateless
 - ✦ Impose no artificial restrictions processing order
- ✦ Processing directed by input and inference reasoning mechanisms

Previous work on declarative approaches

★ Linguistics

- ★ Theories: eg. HPSG, LFG, TAG, CCG
- ★ Older work was purely symbolic and had reputation for being brittle and slow
- ★ More recent work incorporates statistical modeling, but still aims more at parsing than realization, and lacks complete declarativeness

★ Computer Science

- ★ Constraint programming
 - ★ . . .

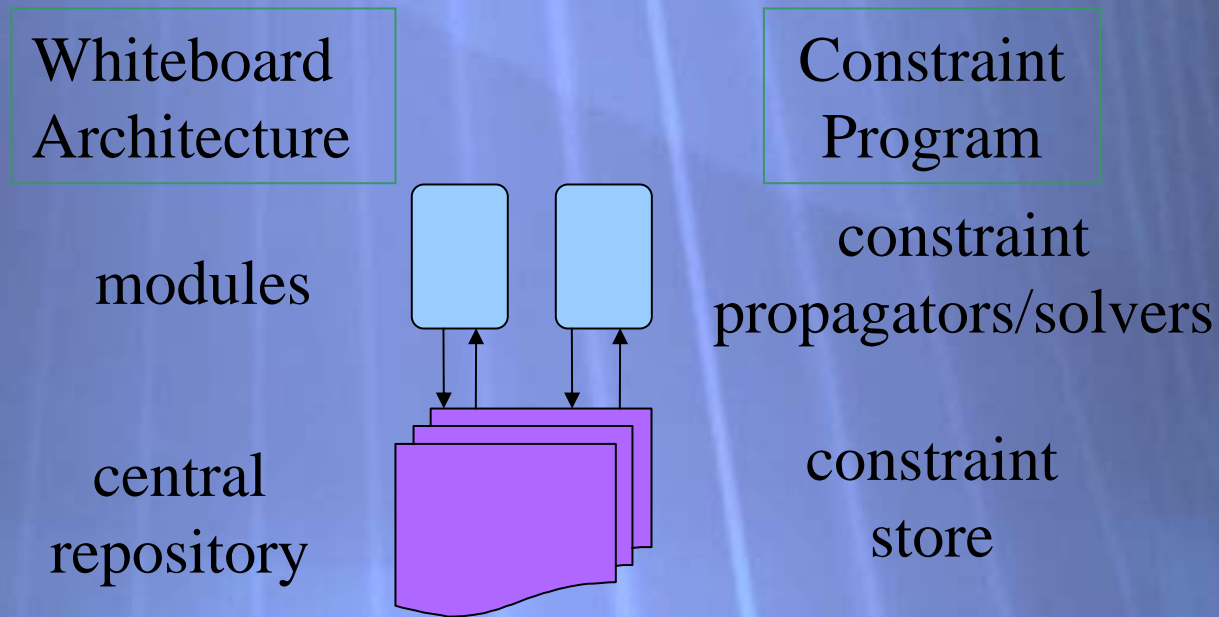
Constraint Programming

- ✦ Roots in
 - ✦ Artificial Intelligence,
 - ✦ Operations Research, and
 - ✦ Programming Language design and implementation
- ✦ Integrates
 - ✦ heterogeneous solvers within
 - ✦ general-purpose methodology having strong theoretical foundations
- ✦ Facilitates high-level model experimentation
- ✦ Designed for finding good average-case solutions to hard combinatorial problems

Our Proposed Approach

- ✦ Concurrent CP as an efficient whiteboard architecture for fine-grained declarativeness
 - ✦ Mozart/Oz language and programming environment
- ✦ Factored dependency-style representation derived from Penn Treebank
 - ✦ Langkilde-Geary and Betteridge, LREC 2006
- ✦ Combination of
 - ✦ hard logical constraint propagation with
 - ✦ soft probabilistic optimization
- ✦ This paper:
 - ✦ Realization and parsing from tokens, not just functional roles
 - ✦ Set-based representation of tree structure

Concurrent CP as a Whiteboard



Modeling with CCP

- ✦ Variables
- ✦ Constraints
 - ✦ Basic: individual variable domains
 - ✦ Non-basic: Relations between variables
- ✦ Search strategy

- ✦ Methodology
 1. Propagate inferences onto var domains
 2. Split into complementary subcases
 3. Repeat

CP Formulation: Variables

FEATURE	VALUE
ID	A word id
HeadID	ID of head word
Token	Inflected word form
Role	Syntactic function
Group type (GT)	Clause, NP, or other
Direction (DIR)	+/- from head
Relative Position (RP)	Tree distance from head
Absolute Position (AP)	Distance from start of sen

CP Formulation (cont): Non-basic Constraints

- ✦ Definitional:
 - ✦ DIR, AP, headID
 - ✦ DIR, RP
 - ✦ DIR and RP of top node
 - ✦ Distinct and sequential sibling RPs (except 0)
 - ✦ Distinct and sequential APs
 - ✦ AP, RP of siblings
- ✦ Projective tree (AP, RP, ID, headID)
 - ✦ uses set constraints
- ✦ Probabilistic dependency scores
 - ✦ $-\log\text{prob}(\text{feature} \mid \text{history})$.

Probabilistic Dependencies

FEATURE	INTERDEPENDENCIES
Group type	head: gt self: role, dir, rp, token
Role	head: role, gt self: gt, dir, rp, token
Dir	self: rp, role, gt
RP	self: dir, role, gt

Example: Node Score

✦ Assume:

✦ Given: token, role

✦ Determination order: dir, rp, gt

NodeScore

$$= \text{FeatScore}(\text{role}) + \text{FeatScore}(\text{dir}) + \\ \text{FeatScore}(\text{rp}) + \text{FeatScore}(\text{gt})$$

$$= \log\text{prob}(\text{role}) + \log\text{prob}(\text{dir}|\text{role}) \\ + \log\text{prob}(\text{rp} \mid \text{role}, \text{dir}, h_{\text{gt}}) \\ + \log\text{prob}(\text{gt} \mid h_{\text{gt}}, \text{role}, \text{dir}, \text{rp})$$

Search Strategy

- ✦ Branch-and-bound search
 - ✦ Sum of feature logprobs is cost to be minimized
- ✦ Case splitting
 - ✦ First stage: split on var whose two most likely values have the greatest difference in likelihood
 - ✦ Second stage: split on var with greatest number of suspensions
- ✦ Ordering
 - ✦ Try values in order of greatest likelihood

Probabilistic Modeling Issues

- ✦ Dynamic conditioning =
finer-grained declarativeness
- ✦ Challenge: combinatorial explosion
- ✦ Alternatives:
 - ✦ Simulate with statically trained model
 - ✦ Marginalizing
 - ✦ Dynamic programming
 - ✦ Train a model for each decomposition
 - ✦ Approximate
 - ✦ Substitute most common value for unknown context feature
 - ✦ Lump conditioning contexts into equivalency classes

Dynamic smoothing (*aka* lazy learning)

- ✦ Ideal for accuracy, but
- ✦ Studied relatively little
- ✦ Slower
- ✦ We are in progress of researching

Sample sentence

Token	Role	GT	AP	RP	Dir	ID	Head
time	sbj	np	1	-1	-	5	2
flies	top	c	2	0	+	2	1
like	rma	o	3	-2	-	7	4
an	det	o	4	-1	-	6	4
arrow	ajt	np	5	1	+	4	2
.	rpunc	o	6	2	+	3	2

Experimental Runs

Given VS. To-Be-Determined	Search Tree Size	Prev. Search Tree Size	Total Search Tree Size	De pth	Num Sols	Best Sol	Num Vars	% Cor rect
token, role, head VS. dir, rp, ap, gt	70	194	2376	8	1	8	24	100
token, role, ap VS. dir, rp, head, gt	71	259	2772	9	2	12	24	100
token, ap VS. role, dir, rp, head, gt	5485	NA	63756	16	2	17	30	97

Search Path to Best Solution: 1st Run (realization)

	Token	Rel	F	Vals	V	Diff
1	time	head	gt	c,np,o	c	0.99
2	an	head	gt	np,o	np	0.99
3	time	self	dir	-,+	-	0.93
4	time	self	gt	np,o	np	0.62
5	arrow	self	dir	+,-	+	0.60
6	arrow	self	rp	1,2	1	0.58
7	an	self	rp	1,2	1	0.22

Search Path to Best Solution: 2nd Run (sort of parsing)

	Token	Rel	F	Vals	V	Diff
1	time	head	gt	c,np,o	c	0.99
2	time	self	gt	np,o	np	0.62
3	arrow	head	gt	o,c	c	0.55
4	an	head	gt	np,o	np	0.99
5	arrow	self	rp	1,2	1	0.58
6	.	head	gt	o,np,c	not o	0.43
7	.	head	gt	c,np	c	0.93
8	like	head	gt	o,np	np	0.43

Search Path to Best Solution: 3rd Run (parsing)

	Token	Rel	F	Vals	V	Diff
1	an	self	rol	det,obj,ajt,subj,clr,rma,lp	det	0.99
2	flies	self	dir	+,-	+	0.82
3	time	self	rol	ajt,subj,rma,tpc	ajt	0.75
4	like	self	rol	rma,ajt,top,tpc,prd,clr,subj	rma	0.72
5	arrow	head	gt	o,c	c	0.55
6	an	head	gt	np,o	np	0.99
7	arrow	self	rp	1,2	1	0.64
8	an	head	rol	ajt,subj	ajt	0.49
9	.	head	gt	o,np,c	Not o	0.43
10	.	head	gt	c,np	c	0.93
11	like	head	gt	o,np	np	0.43

A Happy Marriage of Approaches: **Probabilistic** and **Symbolic**

★ Pros

- ★ Automatically learnable
- ★ Scalable, robust systems
- ★ Graded judgments

★ Cons

- ★ Sparse data/smoothing
- ★ Difficult to distinguish infrequent from impossible

★ Pros

- ★ Strong inferences (efficient)
- ★ Linguistic generalizations

★ Cons

- ★ Labor-intensive
- ★ Combinatorial explosion when rarer events are allowed

Most closely related CP work

✦ Denys Duchier et al. }
✦ Claire Gardent et al. } Active prop., but
no statistics

✦ Tomasz Marciniak et al. }
✦ Dan Roth et al. } ILP

Conclusion

- ✦ A declarative approach is probably the most appropriate way to address the challenges of high-quality realization for MT
- ✦ A hybrid approach combining hard and soft constraints offers computational advantages
- ✦ Need further research to address lazy learning and additional tractability issues