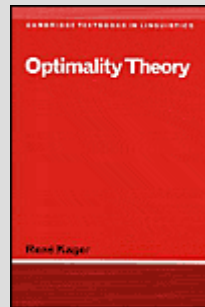
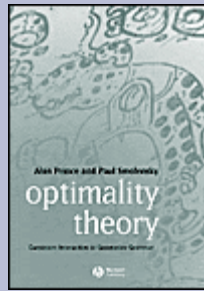
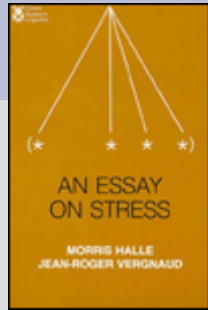


Stress Machine Battlebots

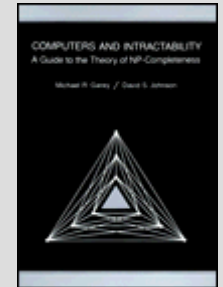
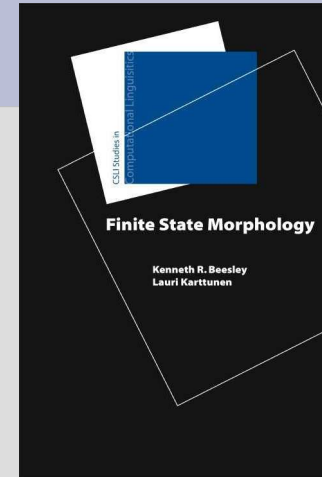
- William Idsardi
- University of Delaware
- idsardi@udel.edu
- <http://www.ling.udel.edu/idsardi/>

- University of Connecticut
- November 19, 2004

Stress



Machine



Overview

- Background
 - Data Structures
 - Algorithms
 - *Direct calculation*
 - *Generate and test*
- Finite state machines
 - *Rules*
 - *Constraints*
- Comparison
 - Battlebots
- Computational complexity

Data Structures

- The representation of information
- Emphasis on:
 - simplicity
 - modularity/encapsulation
 - well-defined interfaces
 - efficiency
- Metrical data structures
 - projection (links to other structures)
 - feet

Algorithms

- Direct procedural calculation
- Generate and test
- “Even ignoring this problem, resolution theorem provers seem to be doomed to using generate-and-test, a very weak method, ...” Charniak and McDermott 1984: 364
- Issues of parallelism

Greatest Common Divisor

- Euclid's algorithm:
 - $\text{gcd}(n,m)$ where $m \leq n$
 - calculate $r = n \text{ modulo } m$
 - if $r = 0$ then answer = m
 - else calculate $\text{gcd}(m,r)$
- example: $\text{gcd}(60,14)$
 - $r = 60 \text{ modulo } 14 = 4$, so $\text{gcd}(14,4)$
 - $r = 14 \text{ modulo } 4 = 2$, so $\text{gcd}(4,2)$
 - $r = 4 \text{ modulo } 2 = 0$, so answer = 2
- sequence of calculations

GCD Generate and Test

- Generate and test:
 - $\text{gcd}(n,m)$ where $m \leq n$
 - for all i $1 \leq i \leq m$ calculate (perhaps in parallel)
 $r_i = n \text{ modulo } i + m \text{ modulo } i$
 - answer = maximum i for which $r_i = 0$
- Example: $\text{gcd}(60,14)$
 - $r_i = \langle 0, 0, 2, 2, 4, 2, 4, 10, 11, 4, 8, 2, 9, 4 \rangle$
 - maximum i for which $r_i = 0$ is 2
- How efficient is each algorithm?
 - count equivalent steps
 - parallel speed-up – size of search space?

OT Evaluation Techniques

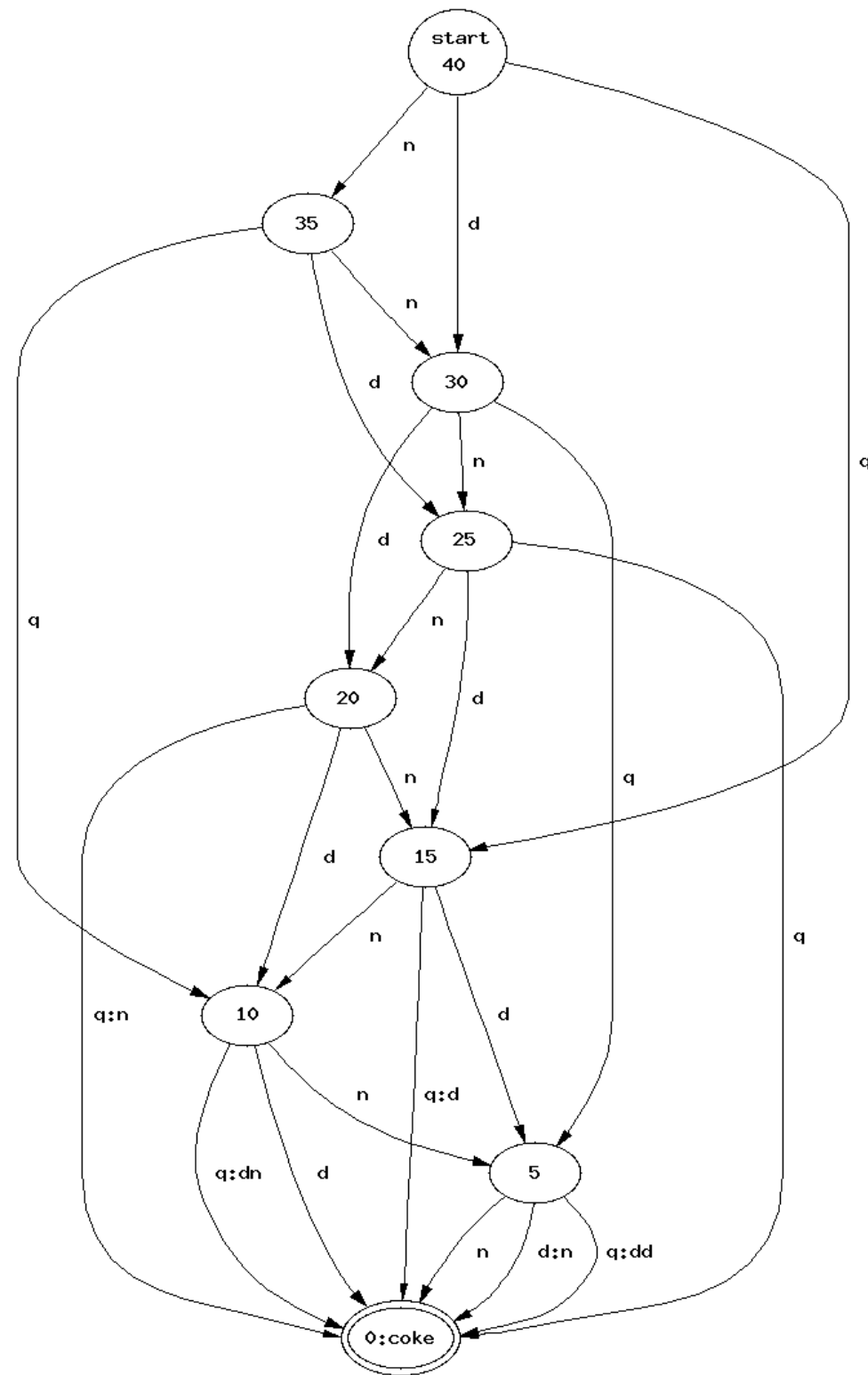
- Parallel evaluation
 - one processor per cell in tableau, i.e. per candidate per constraint
 - parallel sorting techniques to pick winner
 - some problems have an exponential growth rate in candidate space
- Compact representation of candidates
 - Eisner and others: FSAs
 - some intermediate sets of candidates (permutation sets) not efficiently representable

Foot Combinatorics

- For n positions (syllables, moras, ...)
- Exhaustive footing with extrametricality
 - e.g. $\langle x \rangle (xx)(xx) \langle x \rangle = x|xx|xx|x$
 - $2^{(n+1)} - (n+1)$
- Non-exhaustive footing
 - e.g. $x(xx)x(xx)x$
 - $\text{Fib}(2n+1)$
- Exponential growth in problem space
- Run out of processors for parallel generate and test approach

Finite State Machines

- Take input
- Can produce output
Mealey 1955,
Moore 1956
- No extra memory
- Simplest computing device
- Example:
Coke machine
(40 cents)



SBG Overview

- Revisions to the theory
 - Rules and Finite State Machines
 - Parameters and Finite State Machines
 - No Avoidance Constraints
 - No Conflation
 - More use of bracket distinctions, projection
- Typology
- Auca
- Open vs. closed feet

Main theoretical points

- Metrical structure is the counting mechanism
 - Not just stress
 - Tone, infixation, reduplication, feature-docking, poetic scansion, music, ...
- Computationally trivial (Finite State Machine)
- Brackets are real & independent (monopoles)
 - Japanese, Korean tone
 - Sanskrit sonorant retraction
- Bracket distinctions do real work
 - Representational adjunction, theft (open/closed)

Projection

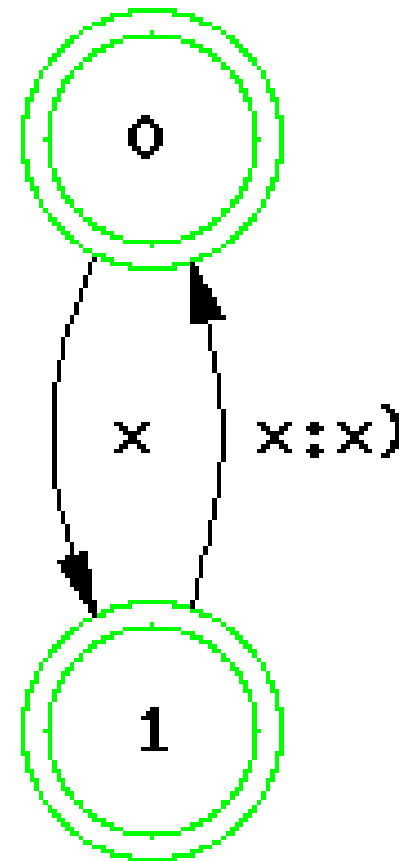
- Create new grid lines
- Each chunk/group/foot/constituent projects:
 - Head: *leftmost* or *rightmost* element
 - Bracket: *left* or *right* or *none*
- All grid planes start at the timing tier (x-tier)
- Projection from x-tier to line 0 can involve syllable properties:
 - e.g. Heavy syllables project (xx)
- Details of syllable depend on theory – only define our side of the interface here (modularity)

Footing

- Bracket insertion:
 - Type of bracket: *left* or *right*
 - Basic foot size: *2* or *3* elements
 - Start scanning from the: *left* or *right*
 - Insert brackets: *iteratively* or *noniteratively*
 - Start with an: *insert* or *skip*
- Replaces both Edge Marking and ICC
 - Edge Marking \approx Non-iterative footing

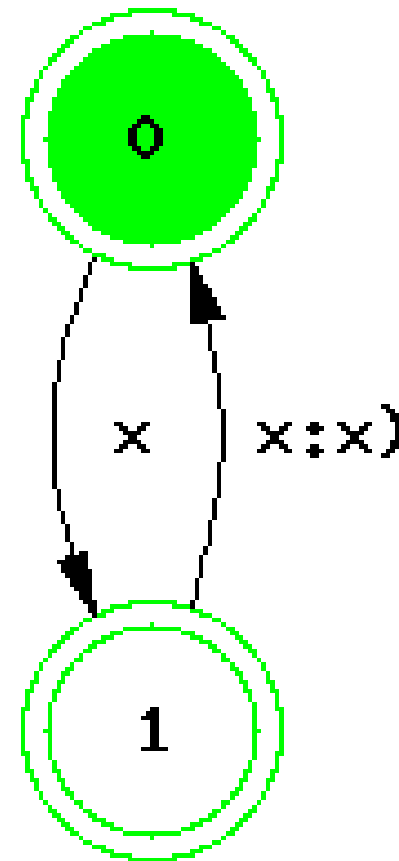
A Footing Machine: R2LIS

- Insert **right** brackets every **2** marks starting the scan from the **left** **iteratively**, starting with a **skip**
- xxxxxxxx
- xx)xx)xx)x



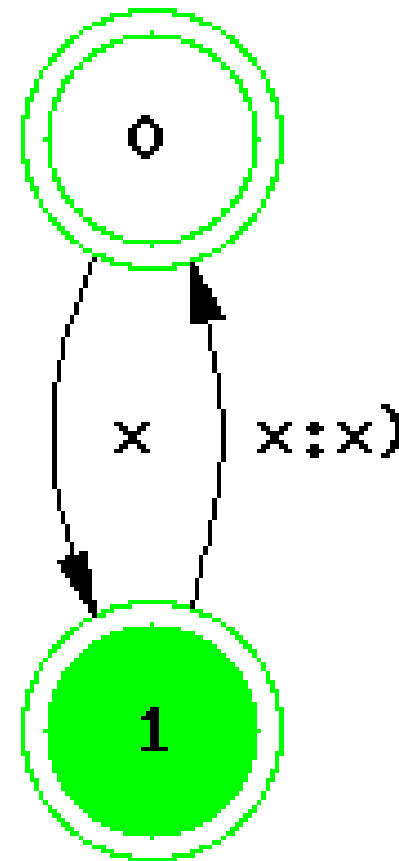
Right 2 Left Iterative Skip - Start

- R2LIS Starting State
- \wedge XXXXXXXX



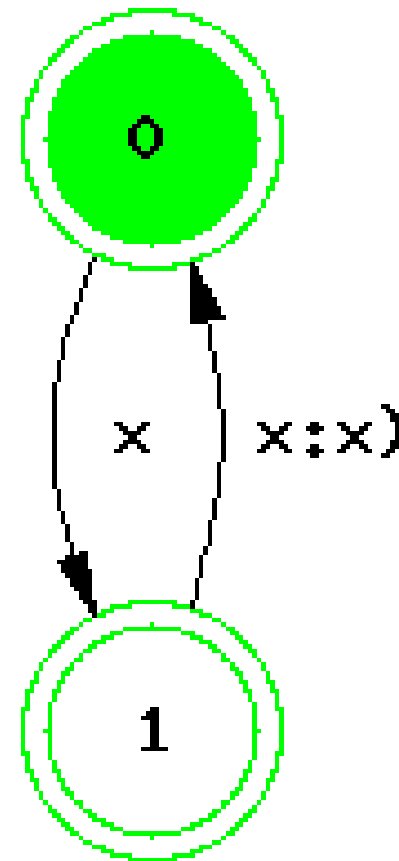
Right 2 Left Iterative Skip

- R2LIS Starting State
- $x \wedge \text{XXXXXXXX}$



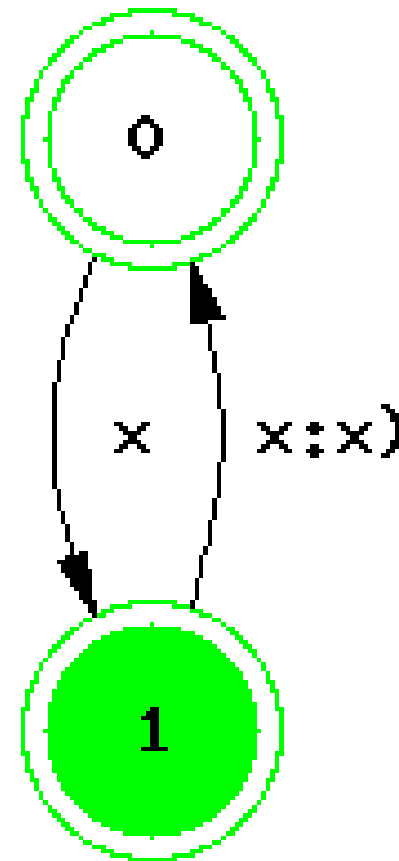
Right 2 Left Iterative Skip

- R2LIS Starting State
- $xx) \wedge xxxxx$



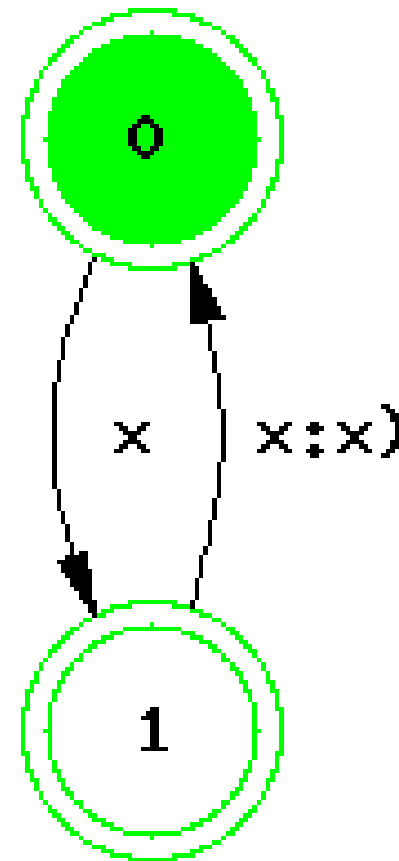
Right 2 Left Iterative Skip

- R2LIS Starting State
- $xx)x_{\wedge}xxxx$



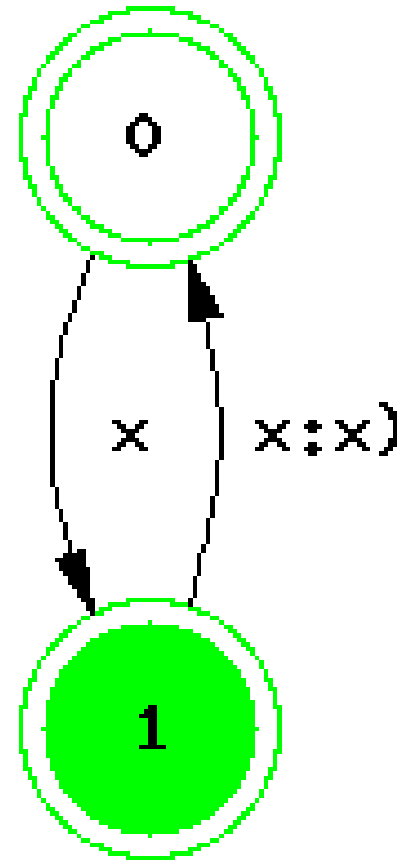
Right 2 Left Iterative Skip

- R2LIS Starting State
- $xx)xx)\wedge xxx$



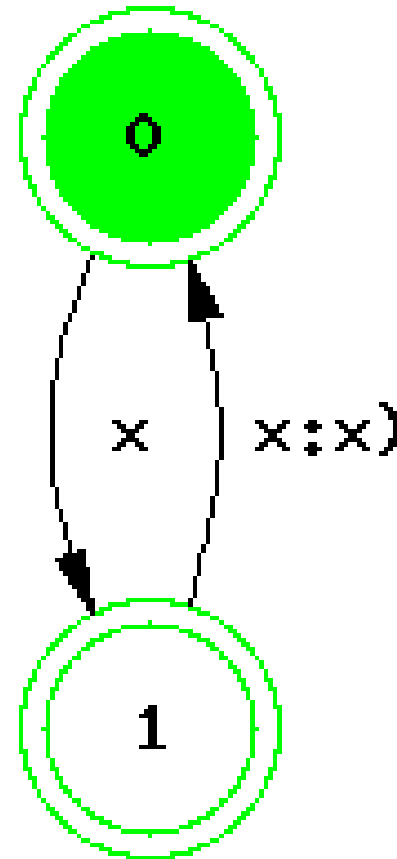
Right 2 Left Iterative Skip

- R2LIS Starting State
- $xx)xx)x^{\wedge}xx$



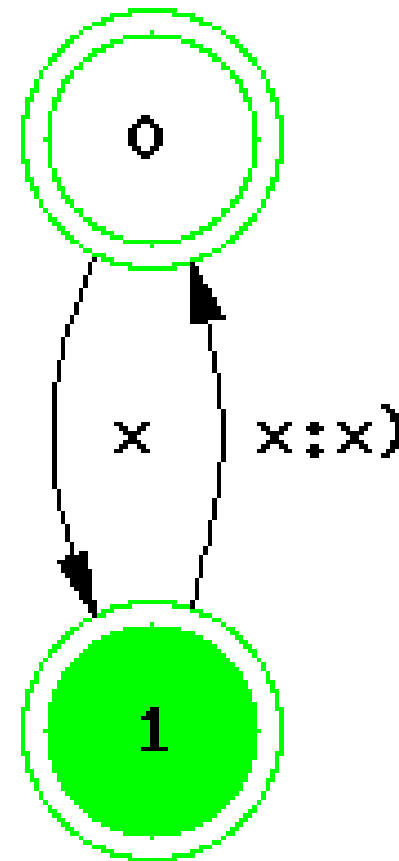
Right 2 Left Iterative Skip

- R2LIS Starting State
- $xx)xx)xx)\wedge x$



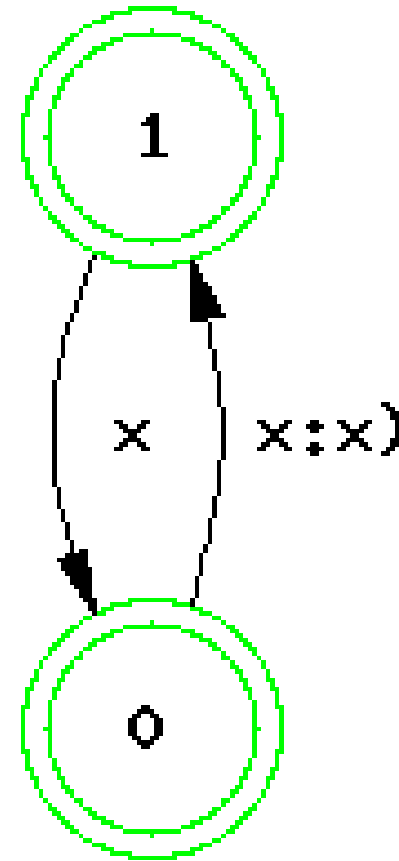
Right 2 Left Iterative Skip - End

- R2LIS Starting State
- $xx)xx)xx)x^{\wedge}$



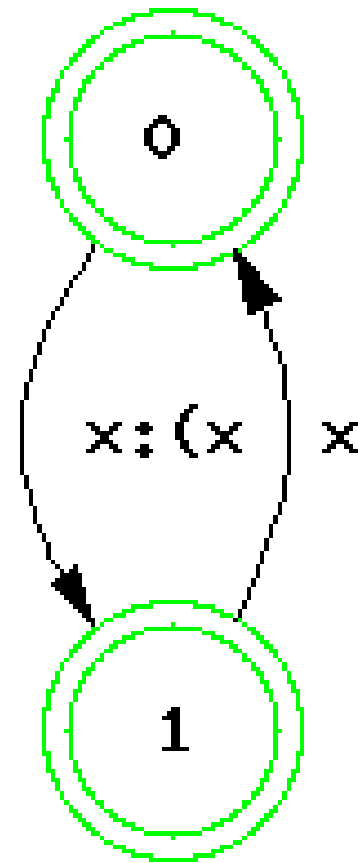
Right 2 Left Iterative Insert

- Same as R2LIS
- Except start in other state
- (By convention start state is numbered zero)
- xxxxxxxx
- x)xx)xx)xx)



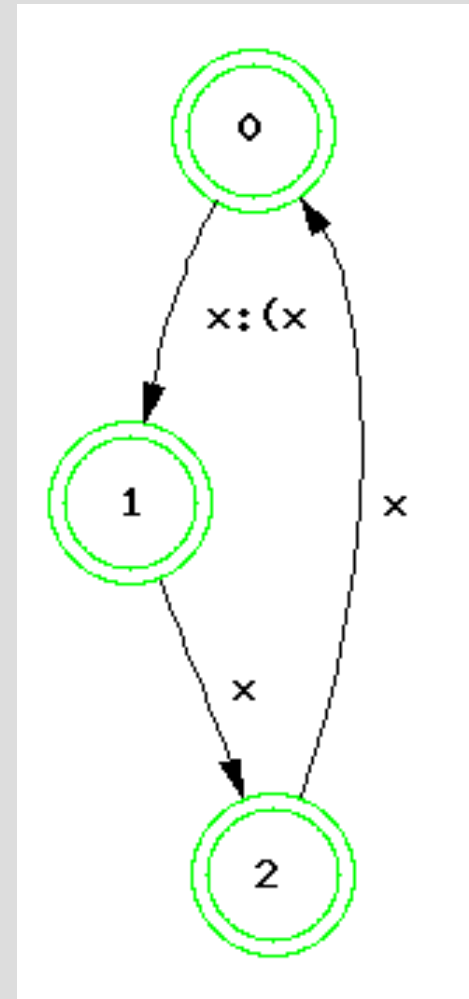
Insert Left brackets

- L2LII
- Same as L2RII except
- Insert Left instead of right
- $(xx(xx(xx$
- $(xx(xx(xx(x$



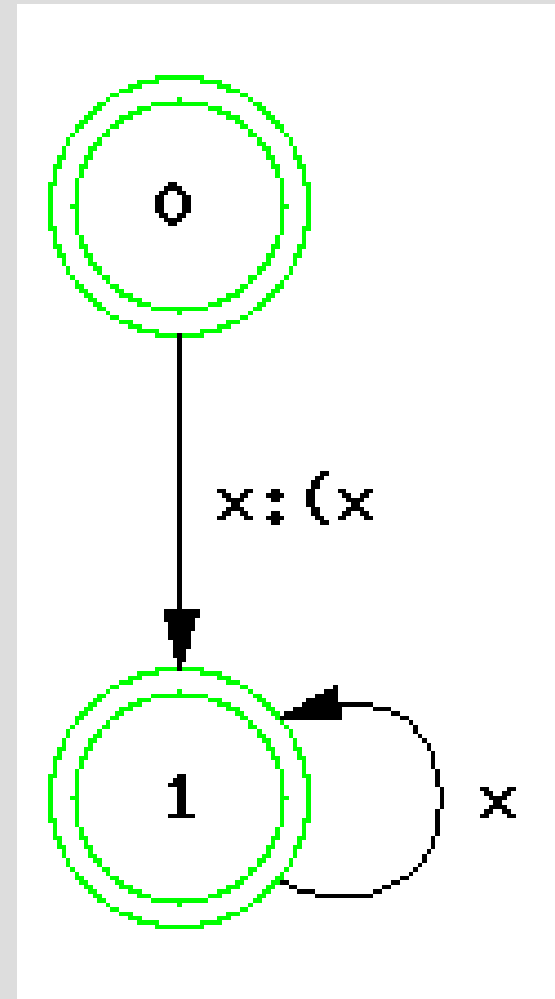
Ternary feet

- L3LII
- Same as L2LII plus
- An extra skip state
 - split state 1
- (xxx(xxx
- (xxx(xxx(x



Noniterative parsing

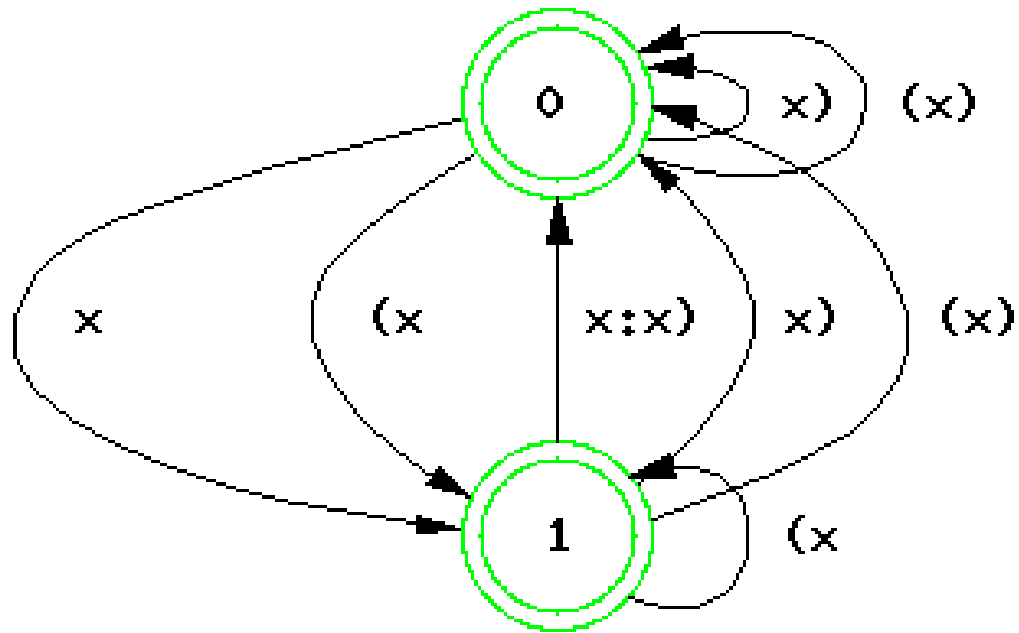
- L2LNI
- Same as L2LII except
- State 1 is a “spin” state
- (xxxxxx
- (xxxxxxxx



Parameters and Machines

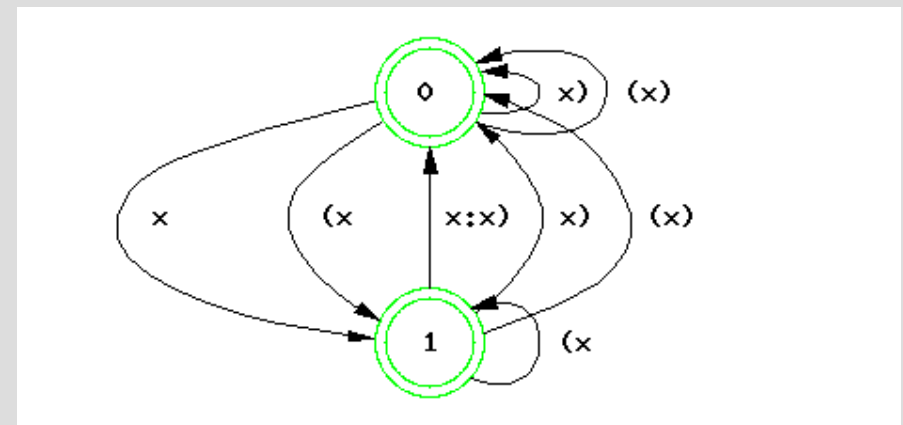
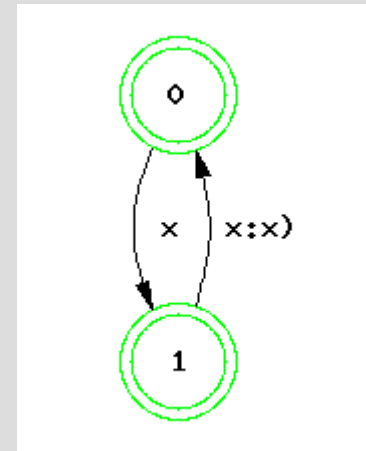
- Start with Basic Machine
 - Insert left or right brackets
- Add an extra skip state
 - Ternary instead of binary parsing
- Add a “spin” state
 - Noniterative instead of iterative
- Change the starting state
 - Start with insert versus skip
 - Edge effects: extrametricality, catalexis
- Each “parameter” can only be done once

Respecting Brackets: R2LIS



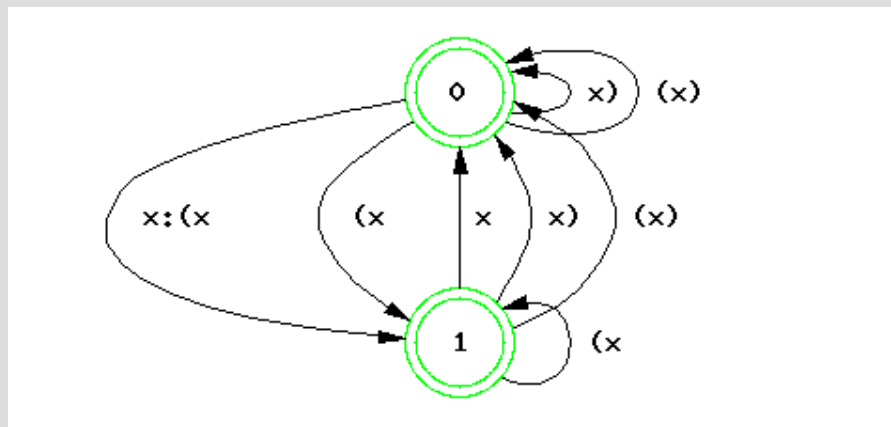
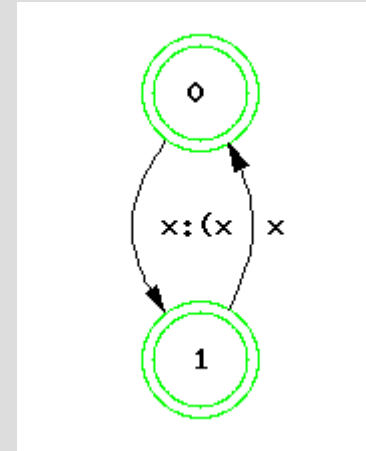
Embellishing Basic Machines

- L2RIS
- Respecting pre-existing brackets is:
- Basic machine plus
 - $(x \rightarrow \text{State } 1$
 - $x) \rightarrow \text{State } 0$
 - $(x) \rightarrow \text{State } 0$



Embellishing Basic Machines

- L2LII
- Basic machine plus
 - $(x \rightarrow \text{State } 1$
 - $x) \rightarrow \text{State } 0$
 - $(x) \rightarrow \text{State } 0$



Partial Typology

Left	Iterative	Insert	(x x(x x(x x	(x x(x x(x x(x	Maranungku?, Auca
Left	Iterative	Skip	x(x x(x x(x	x(x x(x x(x x	Winnebago
Left	Non	Insert	(x x x x x x	(x x x x x x x	North Kyungsang Korean
Left	Non	Skip	x(x x x x x	x(x x x x x x	Tokyo Japanese
Right	Iterative	Insert	x(x x(x x(x	(x x(x x(x x(x	Suruwaha?
Right	Iterative	Skip	(x x(x x(x x	x(x x(x x(x x	Auca suffixes
Right	Non	Insert	x x x x x(x	x x x x x x(x	Turkish?
Right	Non	Skip	x x x x(x x	x x x x x(x x	Indonesian, Polish
Left	Iterative	Insert	x)x x)x x)x	x)x x)x x)x x)	Maranungku?
Left	Iterative	Skip	x x)x x)x x)	x x)x x)x x)x	Araucanian
Left	Non	Insert	x)x x x x x	x)x x x x x x	Tauya
Left	Non	Skip	x x)x x x x	x x)x x x x x	Garawa
Right	Iterative	Insert	x x)x x)x x)	x)x x)x x)x x)	Tauya

Auca

- Collision of two stress trains
 - Stem: odd syllables from left -- Xx...
 - Suffix: even syllables from right -- ...Xx
 - Clash at boundary, e.g. XxX-Xx, XX-xXx
(perfect grid w/o forward-clash-override)
- Stems: L2LII
 - (xx(x-xx
 - (xx-xxx
- Suffixes: L2RIS
 - (xx(x-(xx
 - (x(x-x(xx

Auca observations

- Feet are open on the right
 - forward-clash-override parameter =
bracket inserted
- Suffix train can “steal” a mark to make a foot:
 - $(xx-xxx \rightarrow (x(x-x(xx$
- Respecting metrical structure is directional
 - can steal material on the open side

Other Auca accounts

- Hayes has exceptions to exceptions for his theory:
 - Auca allows sub-minimal feet, e.g. (LL)(L)
 - Revokes sub-minimality to allow stealing:
(xx)(x-x)(xx)
 - Allows non-edge sub-minimal feet in: (x)(x-x)(xx)
- OT account (Kim 2003)
 - stipulates clash occurs in stem, (x)(x-x)(xx) >>
(xx)-(x)(xx)
 - contrary to Align(Stem, Foot)

Open versus closed feet

- Russian /o/ raising
 - (x raises x...) does not
- Sanskrit sonorant retraction
 - pitár-bhis → pitíbhis
 - epenthetic vowel projects inside (_ x
- Old English secondary stress
 - (xx) stronger than xx)
- Japanese pitch accent
 - final accented ...x) > final unaccented (...x
- North Kyungsang Korean accent
 - nonfinal accented ...x)... > final accented (...x

OT and FSA

- Frank and Satta 1998, Karttunen 1998
 - number of violations per constraint must be **bounded**
- But many metrical constraints are gradient:
 - FtBin, Align
 - Could use self-conjunction (Ito & Mester: FtBin²) to mimic gradience with constraint family
- McCarthy 2003: 130
 - “it is sufficient for any constraint to assign one violation-mark for **each instance** of the marked structure or unfaithful mapping in the candidate under evaluation”
- “Each instance” = unbounded

Karttunen

- Karttunen (1998:11):
 - “It is curious that violation counting should emerge as the crucial issue that potentially pushes optimality theory out of the finite-state domain thus making it formally more powerful than rewrite systems...”
- Alternative: constraints as FSMs + sorting
Eval function

Nongradient footing constraints

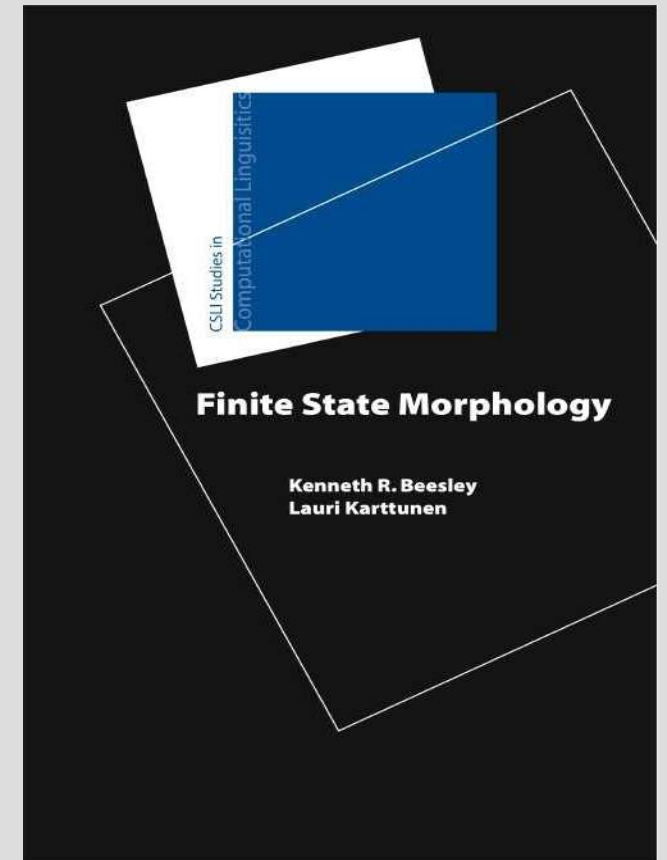
- McCarthy 2003: 106 (following Eisner) proposes a set of replacements for Align
 - *Lapse: assign one violation-mark for each pair of adjacent unstressed syllables
 - Lapse-At-End: «*Lapse» that is not word-final
 - Lapse-At-Peak: «*Lapse» that is not adjacent to the main stressed syllable
 - *InitialLapse: «*Lapse» that is word-initial
- see also Hyde 2002

Left to right parsing

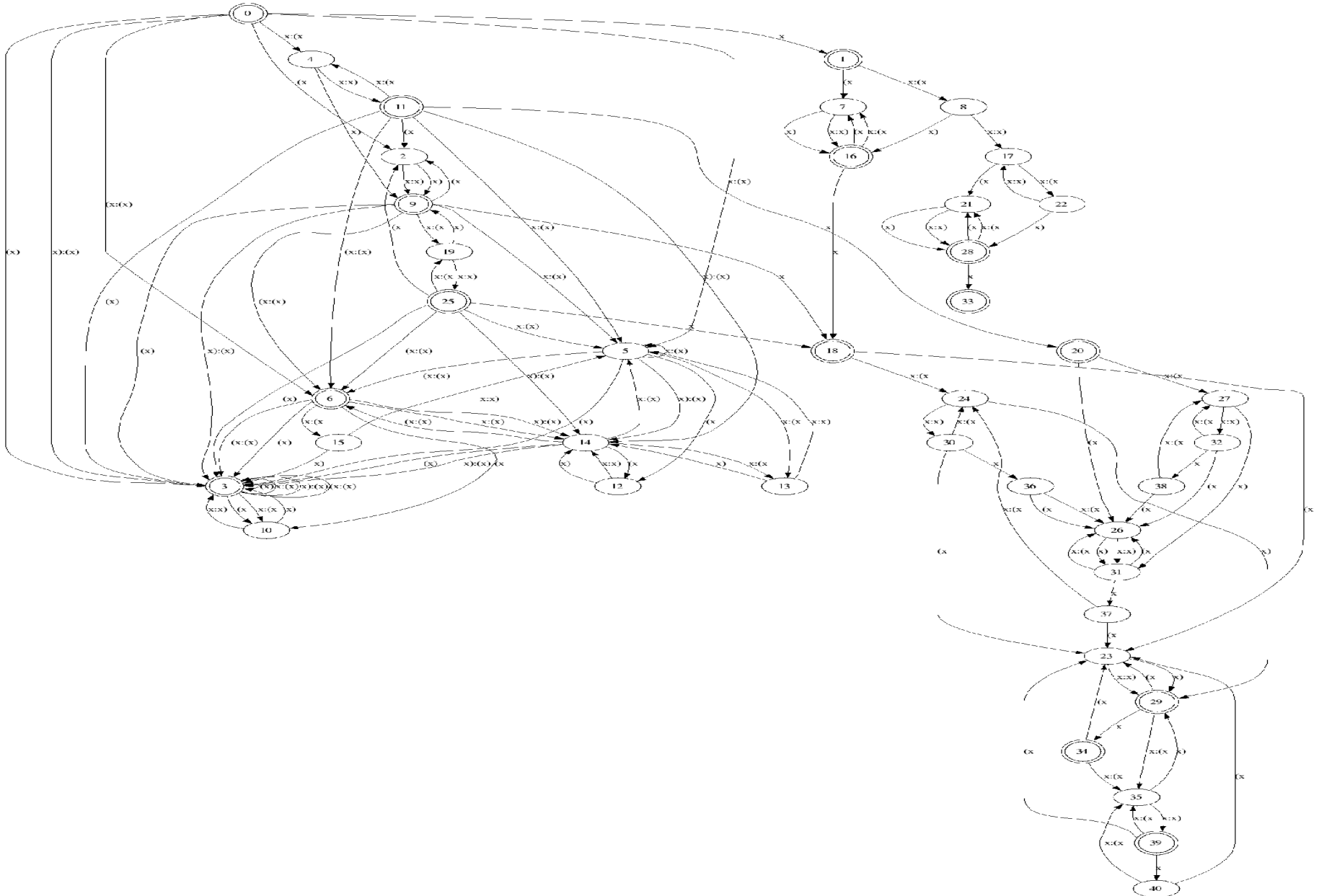
- Footing constraints without headedness
- A set of nongradient footing constraints (in order)
 - FtBinMax: No more than two elements per foot
 - FtBinMin: At least two elements per foot
 - Parse²: No pairs of unparsed elements
 - Parse: No unparsed elements (cf. *Lapse)
 - Unparsed-At-End: No initial or medial unparsed elements (cf. Lapse-At-End)
- Even: (xx)(xx)(xx), Odd: (xx)(xx)(xx)x

Beesley and Karttunen

- Used Beesley and Karttunen 2003 software
- Built network using lenient composition as in Karttunen 1998
- 41 states, 123 arcs
- an order of magnitude larger

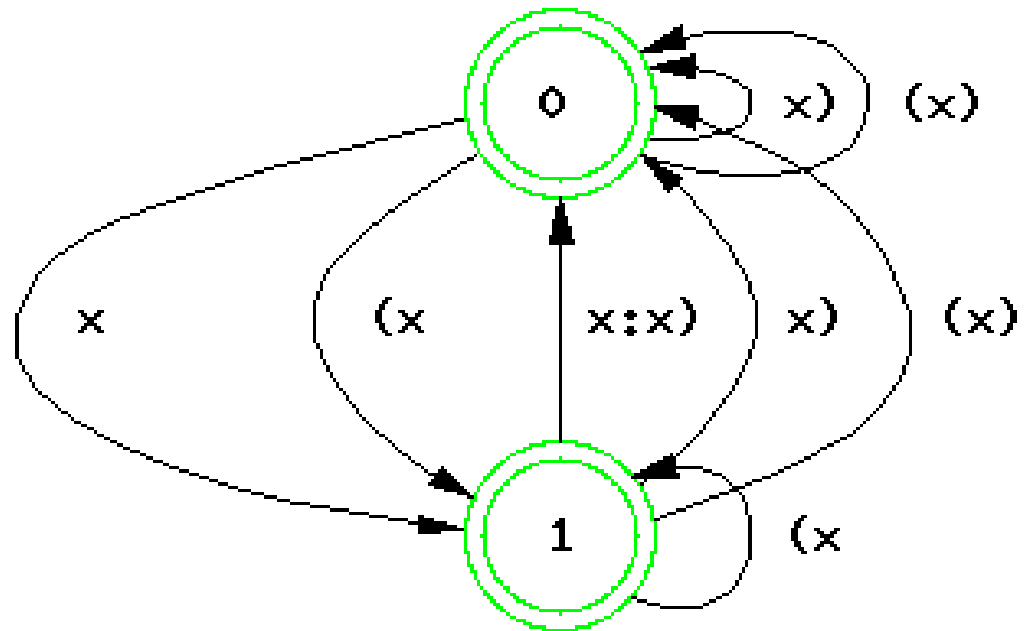
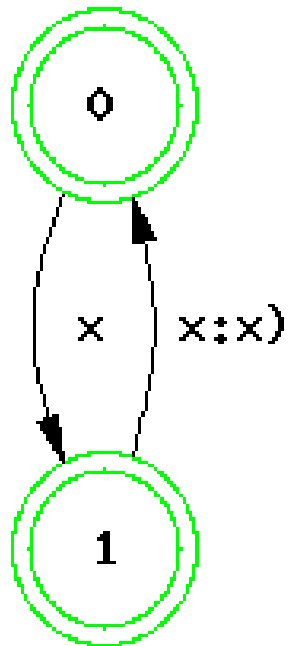


R2LIS in OT



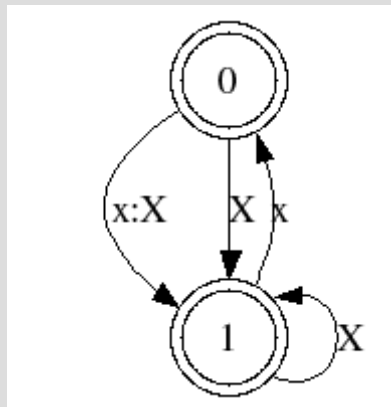
Respecting Brackets: R2LIS

- Insert **right** brackets every **2** marks starting the scan from the **left iteratively**, starting with a **skip**
- Even: $xx)xx)xx)$, Odd: $xx)xx)xx)x$



Grid-only theory

- LtoR perfect grid
- Respect existing stresses
- Rule: 2 states, 4 arcs
- OT: 39 states, 96 arcs

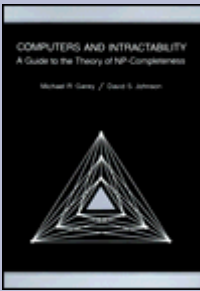


Grammar Battlebots?

- Is FSM a fair battleground?
 - common format for
 - rules
 - constraints
 - Gen
- Creating systems
 - rules: composition
 - constraints: lenient composition
- Is OT network reducable to rule network?
 - non-deterministic FSA minimization is NP-hard (Skiena 1998: 419)

Computational Complexity

- “Hard” problems
- Exponential growth
- NP-complete
- Two kinds of non-results to avoid
 - Ristad: superset is not computable
 - Frank: subset is computable
- Instead either:
 - superset is still computable (good)
 - subset is not even computable (bad)



Traveling Salesman Problem

- see also Eisner
- Given a set of cities visit each city once and only once so as to minimize the cost (or distance) of the trip
- “... perhaps the most notorious NP-complete problem ...” Pemmaraju and Skiena 2003: 303
- “Remains NP-complete even if $d(c_i, c_j) \in \{1, 2\}$ for all $c_i, c_j \in C$.” Garey and Johnson 1979: 211
- If we can encode the TSP in OT, then OT formalism is too powerful

TSP in OT

- Let $A = \{a, b, c, \dots\}$ be the cities
- Consider the UR /abc.../
- Construct CON as follows
 - undominated: Max, Dep, $*\alpha^2$ (Ito and Mester 2003 local self-conjunction, compare Grassmann's Law, Lyman's Law)
 - **net result so far: permutations on A**
 - rank sequence constraints $*ij \gg *mn$ if $\text{cost}(i,j) > \text{cost}(m,n)$ (compare No-NC, etc.)
- Winning candidate = least costly path
 - lexicographic ordering as cost function?



Simple Triangle Example

	Max	Dep	No- α^2	No-ca	No-ac	No-cb	No-bc	No-ab	No-ba
abc							*	*	
acb					*!	*			
bac					*!				*
bca				*!					
cab				*!					
cba						*!			*
abca		*!	*						
ab	*!								
aba			*!						

Hamiltonian Circuit

- Possible objection: lexicographic ordering of \gg is not the same as $>$ over metric distance
- Hamiltonian Circuit (TSP w/o distances) is also NP-complete (Garey and Johnson)
- OT as before, but separate edges with MParse:
- if edge $mn \in G$ and edge $ij \notin G$ then $*ij \gg \text{MParse} \gg *mn$
- all candidates surviving at MParse are Hamiltonian circuits

Conclusions

- Rule systems compile to smaller FSMs than constraint systems (battlebots)
- Natural connection between FSM attributes and footing rule parameters
- Direct calculation is more efficient than generate-and-test
- OT is too powerful, even with McCarthy's restrictions

Kashmiri

- Three degrees of weight:
 - long >> closed >> light
 - The left-most non-final long vowel is stressed, else
 - The left-most non-final closed syllable is stressed, else
 - The left-most syllable is stressed
- Project: light = x, closed = (x, long = (x)
- Nonfinality through R2RNS -- i.e. ...x)x#
- Project: L and) onto line 1
- Project: R onto line 2
- Project: L onto line 3

Kashmiri Examples

- vah.raa.vun 'spread'

- x
x)
x x) x
(x(x)(x

- na.ki.voor 'nostril'

- x
x x)
x) x)
x x)(x

- baa.laa.dər 'balcony'

- x
x x)
x) x) x
(x)(x)(x

- ku.ni.vi.zu 'sometime'

- x
x)
x)
x x x)x

Kashmiri consequences

- Projecting brackets achieves what unmatched bracket deletion did, but in a constructive way
- Gives a theoretical limit to the number of weight classes
- Can be used for other 3-weight systems, including lexical accent with dominance systems (Moses-Columbian, Russian, Sanskrit, Hindi dialects)

Tauya

- Tauya clash similar to Auca (MacDonald)
 - “Primary stress falls on the final syllable...
 - [S]econdary stress on preceding alternate syllables.
 - The initial syllable ... is never without stress...
 - [R]eceives secondary stress, even if this results in adjacent stressed syllables.”
- Initial: R2LNI
- Iterative: R2RII

Tauya schematic examples

- $x \ x \ x$
 $x)xx)xx)$

- $x \ x \ \ \ x \ x$
 $x)x)xx)xx)$

- $x \ x$
 $x)x)$

- Initial: R2LNI
- Iterative: R2RII

Garawa

- Garawa initial dactyls
- Initial: R2LNS
- Iterative: L2RIS
- X X X
xx)(xx(xx
- X X X
xx)x(xx(xx
- Polish has mirror image cases with final trochees: Xx)Xx)x(Xx

Winnebago facts

- All light words: odd non-initial (3, 5, 7, ...)
 - hokiwároké 'swing (n.)'
 - some ternary exceptions
- Initial heavy counts as LL
 - waipéresgá 'linen'
 - heavy projects xx
- Non-initial heavy always stressed
 - hitʔetʔéire
 - heavy project (xx

Winnebago epenthesis

- Dorsey's Law: CvRV is post-accenting
 - wakiripóropóro
 - CvRV = x(x
- Note that epenthetic vowels receive accent
 - Problem for Dep(V)&Dep(Stress) type accounts (Alderete)
- Note also that we can have the first stress four syllables from the left in these cases
 - Also a problem for Alderete's account of stress-epenthesis interaction

Winnebago analysis

- No Avoidance Constraints, no deletion
- Footing: L2LIS, R2LIS
- Project: R,) onto line 1
- Project: R onto line 2
- | | | | | | | | | | | |
|----|-------|----|----|-----|-------|----|----|----|----|----|
| | x | | | x | x | | | | | |
| | x | x) | x | | x | x) | x) | x | | |
| | x(x(x | x) | (x | | x(x(x | x) | (x | x) | (x | |
| wa | ki | ri | pa | ras | wa | ki | ri | po | ro | po |
- i.e. unary feet don't get stress

Chugach Alutiiq

- 2 Iterative footings in Winnebago
- Another case: Chugach Alutiiq
- R2LNS, R3LIS, L2RIS
 - $xx)xxx)xxx) \rightarrow (xX)x(xX)x(xX)$
 - $xx)xxx)xx \rightarrow (xX)x(xX)(xX$
 - $xx)xxx)x \rightarrow (xX)x(xX)x$
- Fortition marks foot-initial consonants

Old English

- Outstanding problem for no-deletion
 - Old English
- Germanic foot (Dresher & Lahiri)
- Initial LH resolves to X.xx
- (→ Ø / (x _ x
- Initial L is stronger than an initial H

Japanese Palatal Prosody

- Like unbounded stress systems
 - Rightmost coronal,
 - else leftmost (noncoronal),
 - but never /r/
- Project:
 - x for onsets,
 - (x for [coronal],
 - nothing for /r/ (like Indonesian schwa)
- R2RNI, Project: L onto line 1
- R2RNI, Project: R onto line 2
- Link [-back] to element linked to line 2

Japanese examples

- meta 'destroyed'

- x
x)
x (x)
meta

- zabu 'dabble in liquid'

- x
x)
(x x)
zabu

- noso 'slowly'

- x
x x)
(x(x)
noso

- horo 'weak'

- x
x)
x)
horo

Applications to Syllables?

- Codas = unfooted
 - e.g. Dell & Elmedlaoui save codas for last
- CV = (CV)
 - Hyman's core mora
- may need a restricted iterative rule:
 - $xx) \rightarrow (xx)$
- hiatus = (CV)V = clash
- Bella Coola reduplication
 - $pwi \rightarrow pupwi$
 - (pu)i at time of reduplication