Augmented Transition Networks

Petr Horáček, Eva Zámečníková and Ivana Burgetová

Department of Information Systems
Faculty of Information Technology
Brno University of Technology
Božetěchova 2, 612 00 Brno, CZ

FRVŠ MŠMT FR97/2011/G1

Outline

- **Introduction**

Outline

- **Introduction**
- **Recursive Transition Network**

Outline

- **Introduction**
- **Recursive Transition Network**
- **Augmented Transition Networks**

Outline

- **Introduction**
- **Recursive Transition Network**
- **Augmented Transition Networks**
- **ATN with Global Register**

Topic

- **Introduction**
- Recursive Transition Network
- Augmented Transition Networks
- ATN with Global Register

Augmented Transition Networks

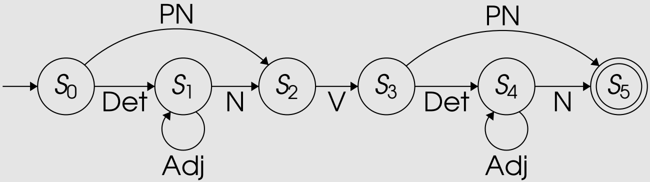
Augmented Transition Networks (ATN)

- One of the formal description of NL.
- One of the earliest implementations of a system for parsing natural languages.
- Work as finite state machines.

Augmented Transition Networks

Definition

- Non-deterministic finite state acceptor (NDA) - quadruple (Q, δ, q_0, F) over alphabet X.
  For a simple fragment of English can be constructed in following way:



- where:
  - X . . . input vocabulary
  - C . . . category vocabulary, made up of grammatical categories of NL (NP – noun phrase, VP – verb phrase, V - verb)
  - function $CAT : X \to C$, such that $CAT(x)$ is the defined category of x.

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

and C is:

$$C = \mathrm{N}, \mathrm{Det}, \mathrm{V}, \mathrm{Adj}, \mathrm{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

and C is:

$$C = \text{N}, \text{Det}, \text{V}, \text{Adj}, \text{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N
- CAT(John) = PN

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

and C is:

$$C = \mathrm{N}, \mathrm{Det}, \mathrm{V}, \mathrm{Adj}, \mathrm{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N
- CAT(John) = PN
- CAT(Mary) = PN

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

and C is:

$$C = \mathrm{N}, \mathrm{Det}, \mathrm{V}, \mathrm{Adj}, \mathrm{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N
- CAT(John) = PN
- CAT(Mary) = PN
- CAT(the) = Det

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

and C is:

$$C = \text{N}, \text{Det}, \text{V}, \text{Adj}, \text{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N
- CAT(John) = PN
- CAT(Mary) = PN
- CAT(the) = Det
- CAT(a) = Det

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

and C is:

$$C = \mathrm{N}, \mathrm{Det}, \mathrm{V}, \mathrm{Adj}, \mathrm{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N
- CAT(John) = PN
- CAT(Mary) = PN
- CAT(the) = Det
- CAT(a) = Det
- CAT(loves) = V

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

and C is:

$$C = \text{N}, \text{Det}, \text{V}, \text{Adj}, \text{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N
- CAT(John) = PN
- CAT(Mary) = PN
- CAT(the) = Det
- CAT(a) = Det
- CAT(loves) = V
- CAT(saw) = V

Example

Example

For example if

$$X = (dog, John, Mary, the, a, loves, saw, white)$$

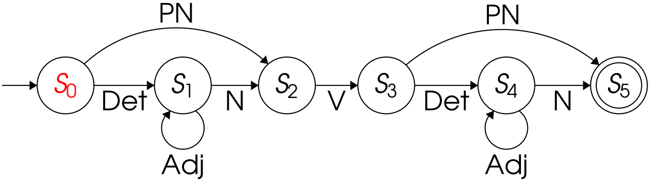
and C is:

$$C = \mathrm{N}, \mathrm{Det}, \mathrm{V}, \mathrm{Adj}, \mathrm{PN}$$

Using the CAT function we define the following relationships:

- CAT(dog) = N
- CAT(John) = PN
- CAT(Mary) = PN
- CAT(the) = Det
- CAT(a) = Det
- CAT(loves) = V
- CAT(saw) = V
- CAT(white) = Adj

Example

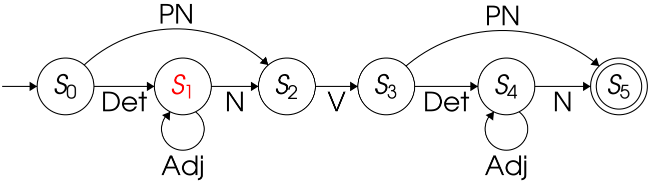


Example

State $\{S_i\}$ is created as follows:

- S_0 . . . initial state

Example

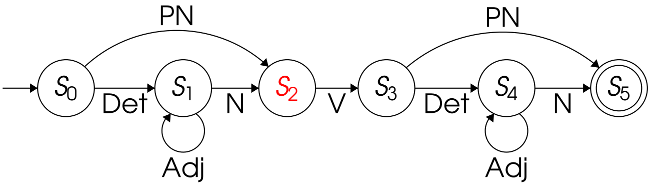


Example

State $\{S_i\}$ is created as follows:

- S_0 ... initial state
- S_1 ... the state in which is legal for the next word to be the noun of the first NP or its adjectives.

Example

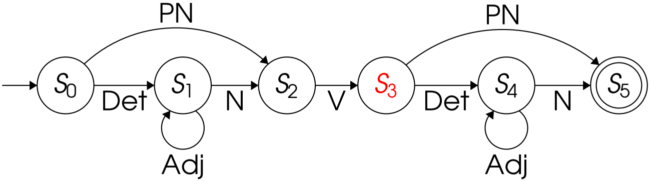


Example

State $\{S_i\}$ is created as follows:

- S_0 . . . initial state
- S_1 . . . the state in which is legal for the next word to be the noun of the first NP or its adjectives.
- S_2 . . . the state reached after reading the first NP.

Example

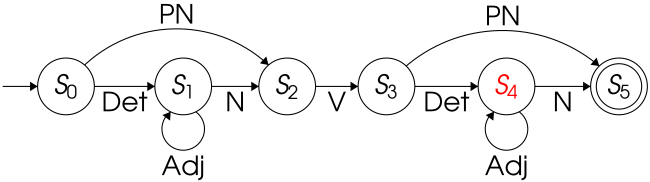


Example

State $\{S_i\}$ is created as follows:

- S_0 … initial state
- S_1 … the state in which is legal for the next word to be the noun of the first NP or its adjectives.
- S_2 … the state reached after reading the first NP.
- S_3 … the state reached after reading of a verb.

Example

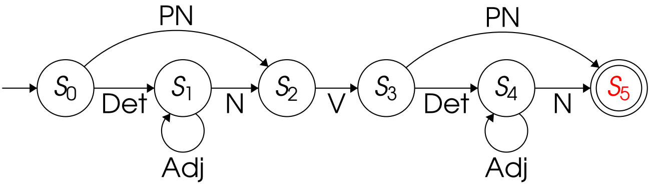


Example

State $\{S_i\}$ is created as follows:

- S_0 . . . initial state
- S_1 . . . the state in which is legal for the next word to be the noun of the first NP or its adjectives.
- S_2 . . . the state reached after reading the first NP.
- S_3 . . . the state reached after reading of a verb.
- S_4 . . . the state in which is legal for the next word to be adjective or the noun of the next NP.

Example



Example

State $\{S_i\}$ is created as follows:

- S_0 . . . initial state
- S_1 . . . the state in which is legal for the next word to be the noun of the first NP or its adjectives.
- S_2 . . . the state reached after reading the first NP.
- S_3 . . . the state reached after reading of a verb.
- S_4 . . . the state in which is legal for the next word to be adjective or the noun of the next NP.
- S_5 . . . completion state and also $F = S_5$

Example

Example

The preceding state diagram corresponds to the following transition function:

- $\delta(S_0, \text{Det}) = \{S_1\}$
- $\delta(S_0, \text{PN}) = \{S_2\}$
- $\delta(S_1, \text{N}) = \{S_2\}$
- $\delta(S_1, \text{Adj}) = \{S_1\}$
- $\delta(S_2, \text{V}) = \{S_3\}$
- $\delta(S_3, \text{Det}) = \{S_4\}$
- $\delta(S_4, \text{N}) = \{S_5\}$
- $\delta(S_4, \text{Adj}) = \{S_4\}$
- $\delta(S_3, \text{PN}) = \{S_5\}$
- for any pair (S, x) not listed $\delta(S, x) = \{S_5\}$

Example

Example

The preceding state diagram corresponds to the following transition function:

- $\delta(S_0, \text{Det}) = \{S_1\}$
- $\delta(S_0, \text{PN}) = \{S_2\}$
- $\delta(S_1, \text{N}) = \{S_2\}$
- $\delta(S_1, \text{Adj}) = \{S_1\}$
- $\delta(S_2, \text{V}) = \{S_3\}$
- $\delta(S_3, \text{Det}) = \{S_4\}$
- $\delta(S_4, \text{N}) = \{S_5\}$
- $\delta(S_4, \text{Adj}) = \{S_4\}$
- $\delta(S_3, \text{PN}) = \{S_5\}$
- for any pair (S, x) not listed $\delta(S, x) = \{S_5\}$

The language accepted by the NDA:

$$L = \{\text{PN} \cup \text{Det Adj} * \text{N}\}.\{\text{V}\}.\{\text{PN} \cup \text{Det Adj} * \text{N}\}$$

Example

Example

The preceding state diagram corresponds to the following transition function:

- $\delta(S_0, \mathrm{Det}) = \{S_1\}$
- $\delta(S_0, \mathrm{PN}) = \{S_2\}$
- $\delta(S_1, \mathrm{N}) = \{S_2\}$
- $\delta(S_1, \mathrm{Adj}) = \{S_1\}$
- $\delta(S_2, \mathrm{V}) = \{S_3\}$
- $\delta(S_3, \mathrm{Det}) = \{S_4\}$
- $\delta(S_4, \mathrm{N}) = \{S_5\}$
- $\delta(S_4, \mathrm{Adj}) = \{S_4\}$
- $\delta(S_3, \mathrm{PN}) = \{S_5\}$
- for any pair (S, x) not listed $\delta(S, x) = \{S_5\}$

The language accepted by the NDA:

$$L = \{\mathrm{PN} \cup \mathrm{Det\ Adj} * \mathrm{N}\}.\{\mathrm{V}\}.\{\mathrm{PN} \cup \mathrm{Det\ Adj} * \mathrm{N}\}$$

Example of accepted sentences:

1. *John loves Mary.*
2. *The white cat saw Mary.*
3. *Mary loves a cat.*

Topic

- Introduction
- **Recursive Transition Network**
- Augmented Transition Networks
- ATN with Global Register

Recursive Transition Network

Problems of NDA formalism

NDA model is unsatisfying:

- The transitions do not capture the effects of constituency (there is no way of stating formally that a path from S_0 to S_2 represents a noun phrase)

Recursive Transition Network

Problems of NDA formalism

NDA model is unsatisfying:

- The transitions do not capture the effects of constituency (there is no way of stating formally that a path from S_0 to S_2 represents a noun phrase)
- FSM fails to capture a significant generalization in the structure of the language.

Recursive Transition Network

Problems of NDA formalism

NDA model is unsatisfying:

- The transitions do not capture the effects of constituency (there is no way of stating formally that a path from S_0 to S_2 represents a noun phrase)
- FSM fails to capture a significant generalization in the structure of the language.
- Any kind of movement dependencies are impossible to capture.

Recursive Transition Network (RTN)

Problems of NDA model - Solution

- Allow state transitions to refer to nonterminals (phrase structures) as well as terminals.

Recursive Transition Network (RTN)

Problems of NDA model - Solution

- Allow state transitions to refer to nonterminals (phrase structures) as well as terminals.
- Allow a transition between two consecutive states in RTN to be made by reading **string** (with NDA is possible to read just one word of category *A* – string *w* belongs to a set *A*)

Recursive Transition Network (RTN)

Problems of NDA model - Solution

- Allow state transitions to refer to nonterminals (phrase structures) as well as terminals.
- Allow a transition between two consecutive states in RTN to be made by reading **string** (with NDA is possible to read just one word of category *A* – string *w* belongs to a set *A*)
- How do we know that *w* belongs to *A*?

Recursive Transition Network (RTN)

Problems of NDA model - Solution

- Allow state transitions to refer to nonterminals (phrase structures) as well as terminals.
- Allow a transition between two consecutive states in RTN to be made by reading **string** (with NDA is possible to read just one word of category *A* – string *w* belongs to a set *A*)
- How do we know that *w* belongs to *A*?
  - Because *w* has been accepted by *another* RTN – that is why they are called recursive.

Recursive Transition Network (RTN)

Problems of NDA model - Solution

- Allow state transitions to refer to nonterminals (phrase structures) as well as terminals.
- Allow a transition between two consecutive states in RTN to be made by reading **string** (with NDA is possible to read just one word of category *A* – string *w* belongs to a set *A*)
- How do we know that *w* belongs to *A*?
  - Because *w* has been accepted by *another* RTN – that is why they are called recursive.
- Recursivity – for recognizing *A* we call *B* which calls *C* etc. until we come to a transition which calls *A*-recognizer again.

Restructuralization of the NDA

Example

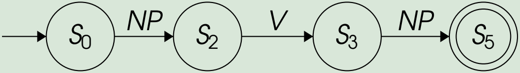
- The first example after restructuralization

Restructuralization of the NDA

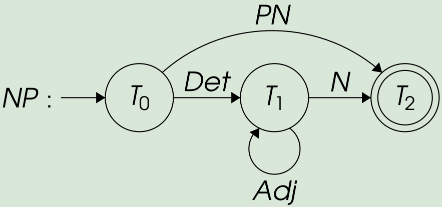
Example

- The first example after restructuralization

The first RTN:



and the second RTN:



Recursive Transition Network

Definition

A recursive transition network R is given by the following data:

1. X is a finite set of *terminal* symbols (eg. words of natural language).
2. C is a finite set of *categories*.
3. CAT is a map $X \to C$.
4. V is a set of *nonterminal* symbols.
5. $S \in V$ is the *start* symbol.

Recursive Transition Network

Recursive Transition Network

- For each $v \in V$, there is a NDA $M_V = \{Q_V, q_v, \delta_V, F_V\}$, where $q_V \in Q_V, F_V \subset Q_V$, and $\delta_V : Q_V \times (C \cup V) \rightarrow 2^Q$
- We associate each $v \in V$ with an acceptance set T_V defined recursively as follows:
  - A string w is in T_V just in case it can be partitioned into a sequence $w_1 w_2 \ldots w_n$ such that either $CAT(w_j) = a_j \in C_j$ or $w_j \in T_{a_j}$ for $a_j \in V$ and such that

$$\delta_v^*(q_V, a_1 a_2 \ldots a_n) \in F_V.$$

Recursive Transition Network

Recursive Transition Network

- For each $v \in V$, there is a NDA $M_V = \{Q_V, q_v, \delta_V, F_V\}$, where $q_V \in Q_V, F_V \subset Q_V$, and $\delta_V : Q_V \times (C \cup V) \to 2^Q$
- We associate each $v \in V$ with an acceptance set T_V defined recursively as follows:
  - A string w is in T_V just in case it can be partitioned into a sequence $w_1 w_2 \ldots w_n$ such that either $CAT(w_j) = a_j \in C_j$ or $w_j \in T_{a_j}$ for $a_j \in V$ and such that

$$\delta_V^*(q_V, a_1 a_2 \ldots a_n) \in F_V.$$

- It means that a string is accepted by M_V with the transitions from states q to q' in M_V either made with a single x with $CAT(x) = c$ such that $q' \in \delta_V(q, c)$ or by a substring w' which is accepted by some M_u for which $q' \in \delta_V(q, u)$.
- The language $L(R)$ accepted by R is just T_S, the acceptance set for the start symbol S.

Recursive Transition Network

Theorem

For every recursive transition network R there is a pushdown automaton (PDA) $P(R)$ which accepts the same language.

Recursive Transition Network

Problems of RTN model

- There is no way to relate *wh*-sentence with its indicative form.
- Some local independencies are impossible to express (subject-verb agreement).

Solution

- RTN can be extended by adding registers to the model.
- Register may hold arbitrary information about input vocabulary.
- Resulting systems are called augmented transition network (ATN).

Topic

- Introduction
- Recursive Transition Network
- **Augmented Transition Networks**
- ATN with Global Register

Augmented Transition Network

Consider two cases involving agreement

- the subject and verb of a sentence
- the determiner and main noun of an NP.

With the RTN (without register) - there is no way to ensure that sentences like the following are prevented from being accepted as a part of the language.

1. ☹The *boys is* mischievous.
2. ☹*A girls* leave home.

Augmented Transition Network

Consider two cases involving agreement

- the subject and verb of a sentence
- the determiner and main noun of an NP.

With the RTN (without register) - there is no way to ensure that sentences like the following are prevented from being accepted as a part of the language.

1. ☹The *boys is* mischievous.
2. ☹*A girls* leave home.

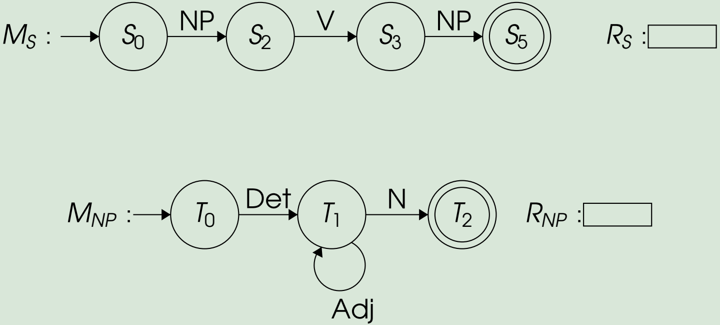
We need to check the number of the subject NP in some way and make the verb transition conditional on the verb agreeing with this NP feature. The following two conditions hold:

Condition I.: If the NP is singular then V must be singular.
Condition II.: If the NP is plural then V must be plural.

Augmented Transition Network: Example

Example

Example of what a register can contain and how this information is structured.



Augmented Transition Network: Example

Example

Each register can hold one of three values:

- singular,
- plural,
- neutral.

Singular does not agree with plural, but neutral agrees with every value.

Augmented Transition Network: Example

Example

Each register can hold one of three values:

- singular,
- plural,
- neutral.

Singular does not agree with plural, but neutral agrees with every value.

Each NDA is augmented as follows:
Transitions for M_S, where w is phrase (or network) variable and x is a word (or arc) variable:

Augmented Transition Network: Example

Example

Each register can hold one of three values:

- singular,
- plural,
- neutral.

Singular does not agree with plural, but neutral agrees with every value.

Each NDA is augmented as follows:
Transitions for M_S, where w is phrase (or network) variable and x is a word (or arc) variable:

Transition 1	$\delta_S(S_0, w) = S_2$ if $w \in T_{NP}$. When transition is made $R_S := n$, where n is value set in R_{NP} by the acceptance of w by M_{NP}

Augmented Transition Network: Example

Example

Each register can hold one of three values:

- singular,
- plural,
- neutral.

Singular does not agree with plural, but neutral agrees with every value.

Each NDA is augmented as follows:
Transitions for M_S, where w is phrase (or network) variable and x is a word (or arc) variable:

Transition 1 $\delta_S(S_0, w) = S_2$ if $w \in T_{NP}$. When transition is made $R_S := n$, where n is value set in R_{NP} by the acceptance of w by M_{NP}

Transition 2 $\delta_S(S_2, x) = S_3$ if $CAT(X) = v$ and the number of x agrees with that stored in R_S.

Augmented Transition Network: Example

Example

Each register can hold one of three values:

- singular,
- plural,
- neutral.

Singular does not agree with plural, but neutral agrees with every value.

Each NDA is augmented as follows:
Transitions for M_S, where w is phrase (or network) variable and x is a word (or arc) variable:

Transition 1	$\delta_S(S_0, w) = S_2$ if $w \in T_{NP}$. When transition is made $R_S := n$, where n is value set in R_{NP} by the acceptance of w by M_{NP}
Transition 2	$\delta_S(S_2, x) = S_3$ if $CAT(X) = v$ and the number of x agrees with that stored in R_S.
Transition 3	$\delta_S(S_3, w) = S_5$ if $w \in T_{NP}$.

Augmented Transition Network: Example

Example

Transitions for M_{NP}:

Transition 4. $\delta_{NP}(T_0, x) = T_1$ if $CAT(X) = Det$. In making the transition, $R_{NP} :=$ the number of x.

Augmented Transition Network: Example

Example

Transitions for M_{NP}:

Transition 4. $\delta_{NP}(T_0, x) = T_1$ if $CAT(X) = Det$. In making the transition, $R_{NP} :=$ the number of x.

Transition 5. $\delta_{NP}(T_1, x) = T_1$ if $CAT(X) = Adj$ and the number of x agrees with that stored in R_{NP}.

Augmented Transition Network: Example

Example

Transitions for M_{NP}:

- Transition 4. $\delta_{NP}(T_0, x) = T_1$ if $CAT(X) = Det$. In making the transition, $R_{NP} :=$ the number of x.
- Transition 5. $\delta_{NP}(T_1, x) = T_1$ if $CAT(X) = Adj$ and the number of x agrees with that stored in R_{NP}.
- Transition 6. $\delta_{NP}(T_2, x) = T_1$ if $CAT(X) = N$ and the number of x agrees with that stored in R_{NP}.

Augmented Transition Network: Example

Example

Transitions for M_{NP}:

Transition 4. $\delta_{NP}(T_0, x) = T_1$ if $CAT(X) = Det$. In making the transition, $R_{NP} :=$ the number of x.

Transition 5. $\delta_{NP}(T_1, x) = T_1$ if $CAT(X) = Adj$ and the number of x agrees with that stored in R_{NP}.

Transition 6. $\delta_{NP}(T_2, x) = T_1$ if $CAT(X) = N$ and the number of x agrees with that stored in R_{NP}.

On exit from T_2, a string w in NP carries the number stored in R_{NP}

Long-distance Dependencies in ATN

Long-distance dependencies

How long-distance dependencies can be handled in an ATN formalism?
Note that neither FSAs nor RTNs were able to parse these sentences (including wh-movement).

1. *What does John love?*
2. *Who loves Mary?*

Long-distance Dependencies in ATN

Long-distance dependencies

How long-distance dependencies can be handled in an ATN formalism?
Note that neither FSAs nor RTNs were able to parse these sentences (including wh-movement).

1. *What does John love?*
2. *Who loves Mary?*

With previous 6 transitions we are now unable to handle such sentences.
One of the possible solutions is to use global register.

Topic

- Introduction
- Recursive Transition Network
- Augmented Transition Networks
- **ATN with Global Register**

Global Register

We need some additional rules.

Global register

- A register which is associated with the whole sentence.

Global Register

We need some additional rules.

Global register

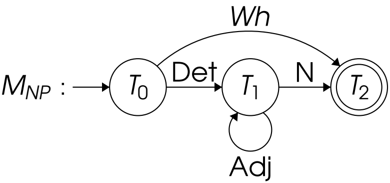
- A register which is associated with the whole sentence.
- Any transition can modify and access this register.

Global Register

We need some additional rules.

Global register

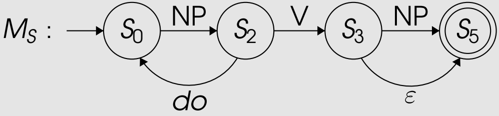
- A register which is associated with the whole sentence.
- Any transition can modify and access this register.
- eg. to parse previous sentences → modify ATN in the following way: we add global register R_{wh} which can be empty, hold *wh* or hold *do*.



We add new transitions to M_{NP}:

Transition 7. $\delta_{NP}(T_0, x) = T_2$ if $CAT(X) = Wh$ in making the transition, $R_{wh} := x$, and $R_{NP} := neutral$.

And to M_S we add these two new transitions:

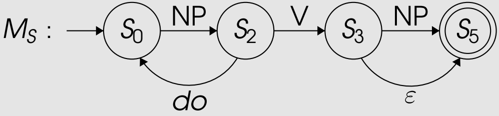


Transition 8. $\delta_M(S_2, x) = S_0$ if $CAT(X) = do$ and $R_{wh} := wh$.
In making the transition, $R_{wh} := do$.

Transition 9. $\delta_M(S_3, \varepsilon) = S_5$ if $R_{wh} := do$.

Global Register

And to M_S we add these two new transitions:



Transition 8. $\delta_M(S_2, x) = S_0$ if $CAT(X) = do$ and $R_{wh} := wh$.
In making the transition, $R_{wh} := do$.

Transition 9. $\delta_M(S_3, \varepsilon) = S_5$ if $R_{wh} := do$.

And to Transition 3 we add the condition $R_{wh} \neq do$

Transition 3 $\delta_S(S_3, w) = S_5$ if $w \in T_{NP}$, $R_{wh} \neq do$.

ATN: Conclusion

This structure accepts both previous sentences 1 and 2 and also sentence 4, but rejects 3 (which is not correct.)

1. What does John love?
2. Who loves Mary?
3. ☹Who likes?
4. Who likes who?

ATN: Conclusion

This structure accepts both previous sentences 1 and 2 and also sentence 4, but rejects 3 (which is not correct.)

1. What does John love?
2. Who loves Mary?
3. ☹Who likes?
4. Who likes who?

The ability to augment a transition with arbitrary tests and computation gives an ATN the computational power of **Turing machines**.

Theorem

For each Turing machine T, there is an ATN which accepts precisely the string accepted by T.

References

James Allen:
Natural Language Understanding,
The Benjamin/Cummings Publishing Company. Inc., 2005

Robert N. Moll, Michael A. Arbib, A. J. Kfoury:
An Introduction to Formal Language Theory,
Springer-Verlag, 1988

Thank you for your attention!

End