University of Bielefeld

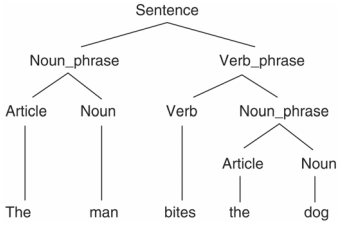
Transition Network Parser

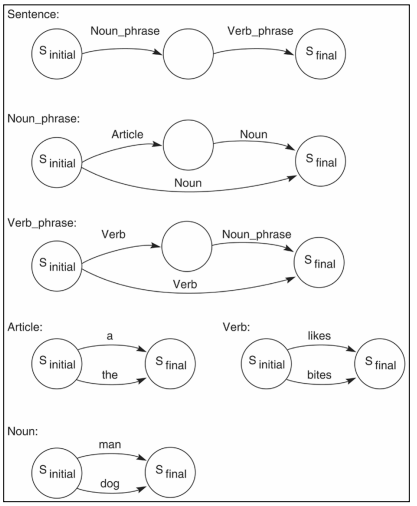
- Grammatik als endlicher Automat oder Übergangsnetzwerk.
- Jedes Netzwerk repräsentiert ein Nichtterminal.
- Kanten repräsentieren Terminale oder Nichtterminale.
- Pfad durch das Netzwerk korrespondiert zu Regel für das entsprechende NT.
- Sequenz der Kanten entspricht der Sequenz der Regeln auf der rechten Seite einer PSG.
- Multiple Regelersetzungen entsprechen multiplen Pfaden im TN.
- Ein erfolgreicher Übergang entspricht einer Ersetzung des entsprechenden NT mit der rechten Seite der Regel.

University of Bielefeld

Transition Network Parser

1.	Sentence	<->	Noun_phrase Verb_phrase
2.	Noun-phrase	<->	Noun
3.	Noun-phrase	<->	Article Noun
4.	Verb-phrase	<->	Verb
5.	Verb-phrase	<->	Verb Noun-phrase
6.	Article	<->	a
7.	Article	<->	the
8.	Noun	<->	man
9.	Noun	<->	dog
10.	Verb	<->	likes
11.	Verb	<->	bites



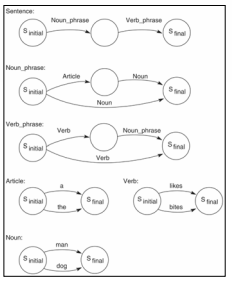


University of Bielefeld

Transition Network Parser

```
function parse(grammar_symbol);

begin
  save pointer to current location in input stream;
  case

    grammar_symbol is a terminal:
      if grammar_symbol matches the next word in the input stream
        then return (success)
        else begin
          reset input stream;
          return (failure)
        end;
    grammar_symbol is a nonterminal:
      begin
        retrieve the transition network labeled by grammar symbol;
        state := start state of network;
        if transition(state) returns success
        then return (success)
        else begin
          reset input stream;
          return (failure)
        end
      end

  end
end.
```

```
function transition (current_state);

begin
  case

    current_state is a final state:
      return (success)
    current_state is not a final state:
      while there are unexamined transitions out of current_state
        do begin
          grammar_symbol := the label on the next unexamined transition;
          if parse(grammar_symbol) returns (success)
                begin
              next_state := state at of the transition;
              if transition(next_state) returns success;
                then return (success)
          end
        end
      return (failure)

  end
end.
```

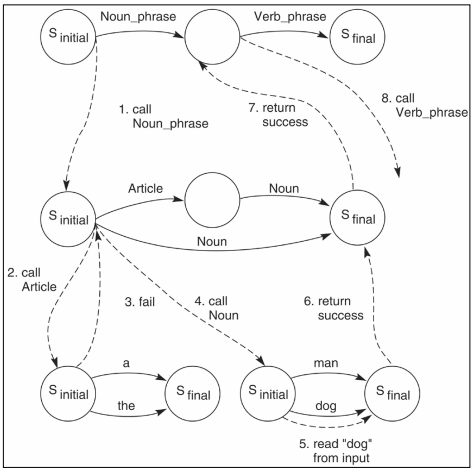


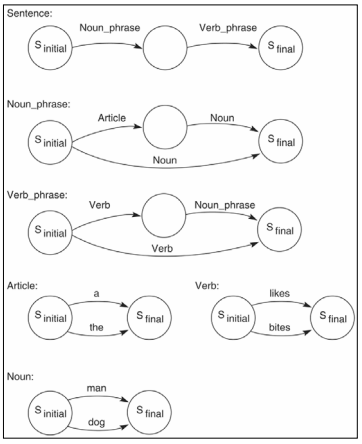
Algorithmus (1) Pseudocode für TNP

University of Bielefeld

Transition Network Parser

Trace des Satzes „Dog bites“:





University of Bielefeld

Transition Network Parser

- Algorithmus (1) prüft Eingabesatz auf Grammatik (t/f) aber generiert keinen parse-tree.
- Parse-tree Generierung kann durch Ersetzung des success Symbols geschehen:
  - Wenn parse(…) mit Terminal aufgerufen wird und dieses Terminal dem aktuellen Inputsymbol entspricht
    dann Rückgabe eines trees mit diesem Symbol.
  - Wenn parse(…) mit NT **grammar_symbol** aufgerufen wird
    dann wird transition(…) aufgerufen.
    Wenn transition(…) erfolgreich war
    dann wird die geordnete Menge subtrees zurückgegeben (s.u.).
    parse(…) kombiniert diese in einen tree mit root **grammar_symbol**.
  - Während der Pfadsuche wird an jedem Kantenlabel parse(…) von transition(…) aufgerufen.
    Wenn Erfolg
    dann Rückgabe des trees des entsprechend geparsten Symbols.
    transition() speichert die subtrees in einer geordneten Menge und gibt während des Pfades die zu den Kantenlabeln korrespondierenden subtrees zurück.

University of Bielefeld

Transition Network Parser

- Welche Sprachklasse nach Chomsky verarbeiten TNPs?
  - ➢ Die der kontextfreien Sprachen.
- Was passiert, wenn wir Rekursion in TNPs verbieten, also die Kantenlabel auf Terminalsymbole beschränken?
  - ➢ TNP wird zu endlichem Automaten reduziert.
  - ➢ Es können nur noch reguläre Ausdrücke verarbeitet werden.
- Für die Verarbeitung verschiedener Phänomene der natürlichen Sprache kann es notwendig sein, Kontext zu berücksichtigen. Wie kann dies geschehen?
  - ➢ Kontext auf der linken Regelseite berücksichtigen.
  - ➢ Annotation.

University of Bielefeld

Kontext berücksichtigen

1.	Sentence		<-> Noun_phrase Verb_phrase
2.	Noun-phrase		<-> Article Number Noun
3.	Noun-phrase		<-> Number Noun
4.	Number		<-> singular
5.	Number		<-> plural
6.	Article singular	<-> a singular	
7.	Article singular	<-> the singular	
8.	Article plural	<-> some plural	
9.	Article plural	<-> the plural	
10.	singular Noun		<-> man singular
11.	singular Noun		<-> dog singular
12.	plural Noun		<-> men plural
13.	plural Noun		<-> dogs plural
14.	singular Verb-phrase		<-> singular Verb
15.	plural Verb-phrase		<-> plural Verb
16.	singular Verb	<-> bites	
17.	singular Verb		<-> likes
18.	plural Verb		<-> bite
19.	plural Verb		<-> like

- singular und plural dienen als Constraints der Regeln. Sie erzwingen Numerus-agreement.
- Gleicher Mechanismus kann für die Berücksichtigung von Semantik dienen (z.B. durch ein NT act_of_biting für bites, so dass man kein Subjekt sein kann.

Probleme:

1. Regelexplosion
2. Phrasenstruktur wird verwässert
3. Vermischung von Syntax und Semantik verliert die Vorteile einer klaren Trennung der syntaktischen und semantischen Sprachkomponenten
4. Es wird keine Semantik aufgebaut sondern nur geprüft

University of Bielefeld

Kontext berücksichtigen

- Ziel:
  - Erhalt der simplen Struktur kontextfreier Grammatik unter Berücksichtigung von Kontextinformationen.
- Möglichkeit:
  - Kontexttests mittels Funktionen/Prozeduren an den Regeln (oder Kanten in RTNs), etwa zum Test von Genus, Numerus und Casus.
- Funktionen greifen auf features zu. Features werden etwa als Register modelliert.
- Funktionen arbeiten als constraints.
- Beispiele:
  - augmented phrase structure grammars [Heidorn, 1975; Sowa, 1984]
  - augmentation of logic grammars [Allen, 1987]
  - augmented transition networks (ATNs)

University of Bielefeld

Augmented Transition Network Parser

- ATNs sind RTNs mit Kantenfunktionen.
- Funktionen/Prozeduren werden während Kantentraversierung ausgeführt.
- Funktionen/Prozeduren
  - können Registerwerte ändern (Numerus, Casus, etc.) und testen
  - konstruieren einen Parsetree
  - generieren eine semantische Repräsentation
- Ts und NTs (z.B. **verb**, **noun_phrase**) werden als Symbole mit assoziierten features dargestellt.
  Für Worte als Terminale werden z.B. folgende Informationen assoziiert:
  - Morphologische Grundform - Wordstamm
  - Wortart
  - Numerus, Casus, Genus

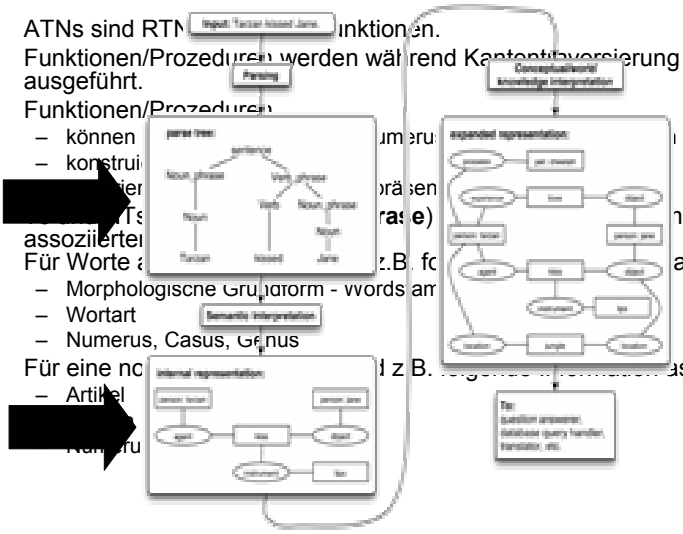
  Für eine noun_phrase als NT wird z.B. folgende Information assoziiert:
  - Artikel
  - Nomen
  - Numerus, Genus

University of Bielefeld

Augmented Transition Network Parser

- ATNs sind RTN[illegible]unktionen.
- Funktionen/Prozeduren werden während Kantentraversierung ausgeführt.
- Funktionen/Prozeduren
  - können [illegible]
  - konstrui[illegible]
- [illegible]Ts[illegible]**rase**)[illegible]mit assoziierten
  Für Worte a[illegible]z.B. fo[illegible]assoziiert:
  - Morphologische Grundform - Words am[illegible]
  - Wortart
  - Numerus, Casus, Genus

  Für eine no[illegible]d z.B. [illegible]assoziiert:
  - Artikel
  - [illegible]



University of Bielefeld

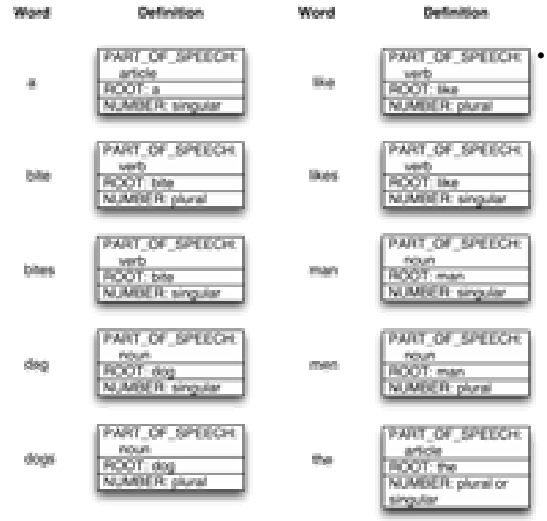
Augmented Transition Network Parser

- Ts und NTs können als frames mit benannten slots und values dargestellt werden.
- slotvalues sind entweder features oder pointer zu anderen frames.

- Einzelne Worte werden mit den gleichen Strukturen im Lexikon eingetragen.

University of Bielefeld

Augmented Transition Network Parser



- Beispiel frames für die Terminale der einfachen dogs-world Grammatik:
  - Es wird nur auf Numerus agreement getestet (weiter features analog).
  - Diese Lexikoneinträge können auch auf eine semantische Repräsentation –etwa einen conceptual graph- abbilden oder mit diesem verbunden sein.

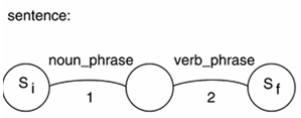
University of Bielefeld

Augmented Transition Network Parser

- Parsevorgang:
  - Für jedes aufgerufene NT wird der entsprechende frame instanziiert.
    - Die slots dieses frames werden von den entsprechenden Funktionen gefüllt.
    - Als Rückgabewert wird der frame geliefert.
  - Wenn das NT eine Wortart repräsentiert
    - wird das nächste Wort vom Input gelesen.
    - wird die Wortdefinition aus dem Lexikon geholt.
    - Wenn das Wort passt, so wird der Frame zurückgeliefert.
    - Wenn das Wort nicht passt wird failure geliefert.
- ➢ Was soll gemacht werden, wenn nicht alle slots gefüllt werden können?

University of Bielefeld

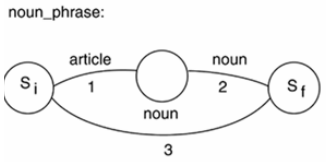
Augmented Transition Network Parser




```
function sentence-1;
  begin
    NOUN_PHRASE := structure returned by
          noun_phrase network;

    SENTENCE.SUBJECT := NOUN_PHRASE;
  end.

function sentence-2;
  begin
    VERB_PHRASE := structure returned by
          verb_phrase network;

    if NOUN_PHRASE.NUMBER =
                        VERB_PHRASE.NUMBER
      then begin
        SENTENCE.VERB_PHRASE := VERB_PHRASE;
        return SENTENCE
      end
      else fail
  end.
```

University of Bielefeld

Augmented Transition Network Parser



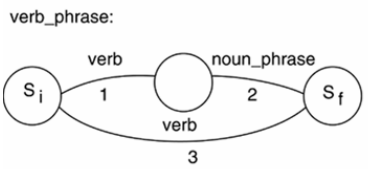

```
function noun_phrase-1;
  begin
    ARTICLE := definition frame for next word of input;

    if ARTICLE.PART_OF_SPEECH=article
      then NOUN_PHRASE.DETERMINER := ARTICLE
      else fail
  end.
```

```
function noun_phrase-2;
  begin
    NOUN := definition frame for next word of input;

    if NOUN.PART_OF_SPEECH=noun and
          NOUN.NUMBER agrees with
               NOUN_PHRASE.DETERMINER.NUMBER
      then begin
        NOUN_PHRASE.NOUN := NOUN;
        NOUN_PHRASE.NUMBER := NOUN.NUMBER
        return NOUN_PHRASE
      end
      else fail
  end.
```

```
function noun_phrase-3
  begin
    NOUN := definition frame for next word of input;

    if NOUN.PART_OF_SPEECH=noun
      then begin
        NOUN_PHRASE.DETERMINER := unspecified;
        NOUN_PHRASE.NOUN := NOUN
        NOUN_PHRASE.NUMBER := NOUN.NUMBER
      end
      else fail
  end.
```


University of Bielefeld

Augmented Transition Network Parser




```
function verb_phrase-1
  begin
    VERB := definition frame for next word of input;

    if VERB.PART_OF_SPEECH=verb
      then begin
        VERB_PHRASE.VERB := VERB;
        VERB_PHRASE.NUMBER := VERB.NUMBER;
      end;
  end.
```

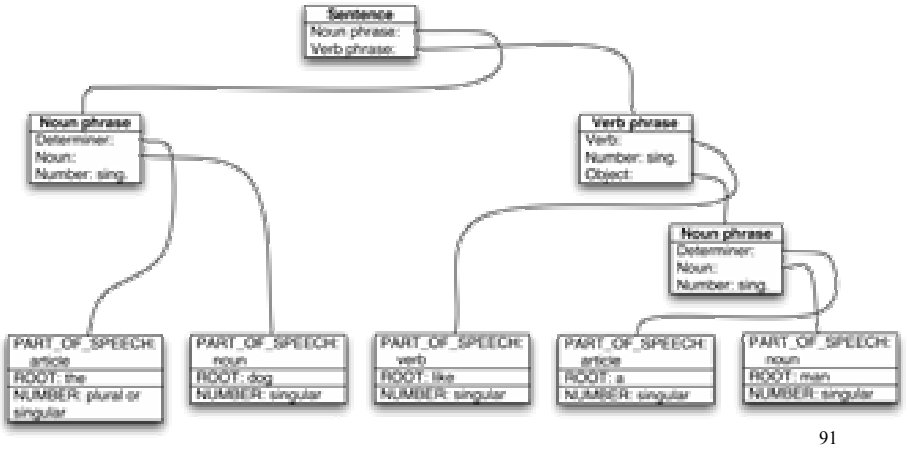
```
function verb_phrase-2
  begin
    NOUN_PHRASE := structure returned by
                          noun_phrase network;

    VERB_PHRASE.OBJECT := NOUN_PHRASE;
    return VERB_PHRASE
  end.
```

```
function verb_phrase-3
  begin
    VERB := definition frame for next word of input;

    if VERB.PART_OF_SPEECH=verb
      then begin
        VERB_PHRASE.VERB := VERB;
        VERB_PHRASE.NUMBER := VERB.NUMBER;
        VERB_PHRASE.OBJECT := unspecified;
        return VERB_PHRASE;
      end;
  end.
```

University of Bielefeld

Augmented Transition Network Parser

ATN-Parser parse-tree für den Satz „The dog likes a man.



University of Bielefeld

Augmented Transition Network Parser

Semantische Interpretation:

- Semantischer Interpreter arbeitet auf dem erzeugten parse-tree.
- Vom root aus wird rekursiv der Baum traversiert; die Teilergebnisse der Rekursionsschritte werden zurückgegeben und kombiniert -> join-Operation.
- Rekursion stoppt an den Terminalen. Terminale wie Nomen, Verben oder Adjektive veranlassen eine Konzeptquery aus der Wissensbasis. Andere Sprachpartikel qualifizieren andere Konzepte durch die syntaktische Struktur.

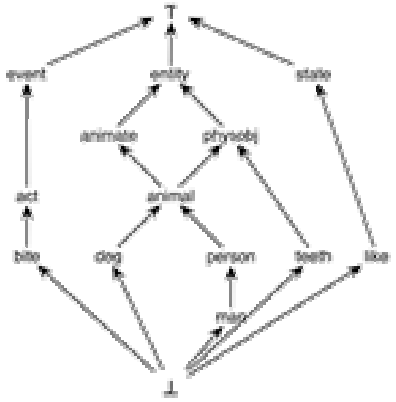
➢ Benötigt wird eine Wissensbasis mit

- Abbildung von lexikalischen Wortarten auf Konzepte und
- Relationen zwischen den Konzepten für den Aufbau des semantischen Struktur, hier am Beispiel eines conceptual graphs.

University of Bielefeld

Augmented Transition Network Parser

Typenhierarchie für die dogs-world:



Weitere benötigte Konzepte und Relationen:

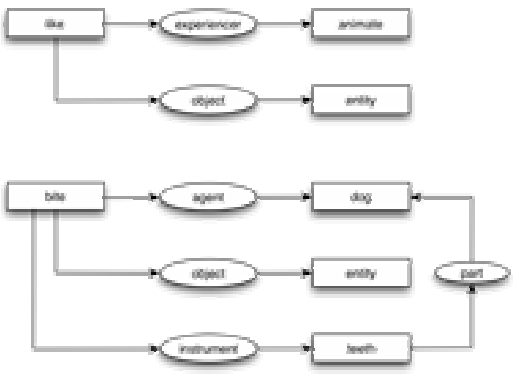
- **agent** verbindet **act** mit **animate** und definiert die Relation zwischen Aktion und Quellobjekt.
- **experiencer** verbindet **state** mit **animate** und definiert für Ziel einen mentalen Zustand.
- **instument** verbindet **act** mit **entity** und definiert ein Instrument einer Aktion.
- **object** verbindet **event** oder **state** mit **entity** und repräsentiert die Verb-Objekt Relation.
- **part** verbindet **physobj**-Konzepte und definiert Ganzes-Teil Beziehungen.

University of Bielefeld

Augmented Transition Network Parser

- Verben sind zentral für die semantische Interpretation, sie bestimmen den Zusammenhang zwischen Subjekt, Objekt und anderen Satzkomponenten.
- Spezielle Rahmen-frames (RF) spezifizieren
  - den linguistischen Zusammenhang des entsprechenden Verbs.
  - Constraints für bestimmte RF-Komponenten.
  - Defaults für bestimmte RF-Komponenten.

RFs für like und bite:



University of Bielefeld

Augmented Transition Network Parser

- Semantische Repräsentation wird durch Prozeduren oder Regeln an jedem potentiellem Parse-tree Knoten aufgebaut. Wenn join- oder test-Operationen misslingen, so ist die Eingabe semantisch inkorrekt.

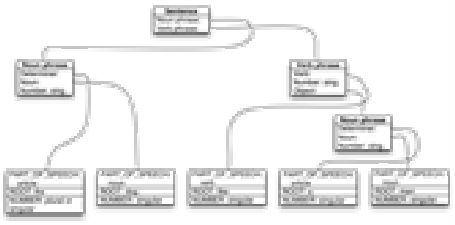
```
procedure sentence;

  begin
    call noun_phrase to get a representation of the subject;
    call verb_phrase to get a representation of the verb_phrase;
    using join and restrict, bind the noun concept returned for the subject to
      the agent of the graph for the verb_phrase
  end.


procedure noun_phrase;

  begin
    call noun to get a representation of the noun;
    case
      the article is indefinite and number singular: the noun concept is generic;
      the article is definite and number singular: bind marker to noun concept;
      number is plural: indicate that the noun concept is plural
    end case
  end.
```

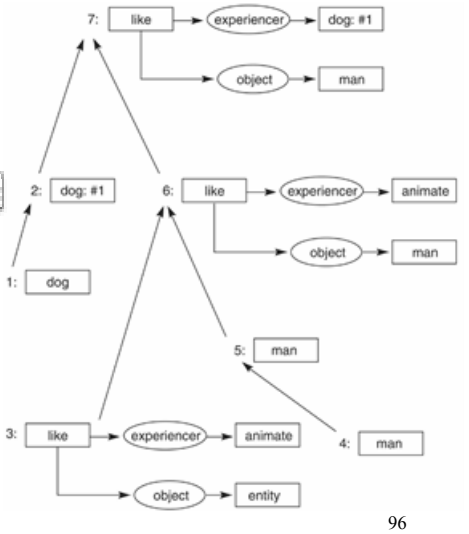
```
procedure verb_phrase;

  begin
    call verb to get a representation of the verb;
    if the verb has an object
      then begin
        call noun_phrase to get a representation of the object;
        using join and restrict, bind concept for object to object of the verb
      end
    end.

procedure verb;

  begin
    retrieve the case frame for the verb
  end.

procedure verb;

  begin      noun;
    retrieve tne concept for the noun
  end.
```

- Keine Konzepte für Artikel, sie bestimmen nur ob DNP oder INP.
- Ein Überblick in Numerus-Modellierung s. [Sowa, 1984].

University of Bielefeld

Augmented Transition Network Parser





- Start am **sentence** Knoten.
- **sentence** ruft **noun_phrase** auf.
- **noun_phrase** ruft **noun** auf.
- **noun** liefert Konzept für das Nomen **dog** (1).
- Wegen definitem Artikel bindet **noun_phrase** einen Individuumsmarker an das Konzept (2) und liefert es an **sentence**.
- **sentence** ruft **verb_phrase** auf.
- **verb_phrase** ruft **verb** auf, dieses liefert den RF für **like** (3)
- **verb_phrase** ruft **noun_phrase** auf, dieses ruft **noun** auf und erhält das Konzept für **man** (4).
- Wegen indefinitem Artikel bleibt **noun** Konzept generisch (5).
- Die **verb_phrase**-Prozedur „restricts“ das **entity**-Konzept im RF und „joins“ diesen mit dem **man**-Konzept (6).
- **sentence** „joins“ Konzept **dog:#1** in den **experiencer**-Knoten des RF (7).