Conceptual Dependency Theory

- Conceptual Dependency (CD) [Schank 1975] is a theory for deep, conceptual language understanding.
  - focus on concepts rather than syntax
  - focus on understanding rather than structure
  - views inference as being fundamental to language understanding

- CD is a canonical meaning representation
  - different words and structures with the same meaning should have the same representation
  - the representation should be language-independent

Motivation for Canonical Meaning Representations

John gave Mary a book.
John gave a book to Mary.
Mary was given a book by John.
A book was given to Mary by John.
Mary received a book from John.
John donated his book to Mary.

Conceptual Primitives for Actions

Eleven primitives are the basis for most activities:

ATRANS
ATTEND
EXPEL
GRASP
INGEST
MBUILD
MOVE
MTRANS
PROPEL
PTRANS
SPEAK
Primitives for Physical Actions

**INGEST**: to take something inside an animate object

**EXPEL**: to take something from inside an animate object and force it out

**GRASP**: to physically grasp an object

**MOVE**: to move a body part

**PROPEL**: to apply a force to something

Other Primitives

State Changes

**PTRANS**: to change the location of an object

**ATRANS**: to change the abstract relationship of an object

Mental Acts

**MTRANS**: to transfer information mentally

**MBUILD**: to create or combine thoughts internally

Senses, often instruments for other Acts

**SPEAK**: an animate object producing a sound

**ATTEND**: an animate object directing a sense organ toward a stimulus

Conceptual Syntax Rules

\[
\begin{align*}
PP & \leftrightarrow ACT \\
PP & \leftrightarrow PA \\
ACT & \leftrightarrow^o PP \\
ACT & \leftrightarrow D LOC \\
ACT & \leftrightarrow R PP \\
\end{align*}
\]

---

An ACT can have an agent.

(PP = Picture Producer)

A PP can have an attribute.

(PA = Picture Aider)

An ACT can have an object.

An ACT can have a direction.

An ACT can have a recipient.

---

An object can be a conceptualization.

An instrument can be a conceptualization.

A PP can be a conceptualization.

An conceptualization can have a time.

An conceptualization can have a location.
Conceptual Syntax Rules

A conceptualization can result in a state change.

A state change can enable a conceptualization to occur.

A mental act can be a reason for a conceptualization.

A PP can be equivalent to another PP.

An ACT can have a quality.

States

States of objects are described using numerical values.

For example:

| HEALTH          | RANGE: |-10 to 10 |
|-----------------|--------|
| dead            | -10    |
| gravely ill     | -9     |
| sick            | -9 to -1 |
| under the weather | -2    |
| all right       | 0      |
| tip top         | +7     |
| perfect health  | +10    |

John went to New York.

John sold his car to Bill.

Mary cried.

John sold his car to Bill.
John annoyed Mary.

John killed Mary.

John killed Mary by throwing a rock at her.

John read a book.
Primitive ACTs facilitate inference

- General inferences are stored with each primitive.
- Reduces the number of inferences that need to be stored explicitly.
- A set of primitives is only as good as the inferences that they generate.
- The meaning of the concept consists of the meaning of the primitive itself and its associated inferences.

What is an inference?

An inference is a conceptualization that is spawned by another conceptualization with a (usually high) probability < 1.

The “but not” test:

- If it makes sense to say “X but not Y”, then Y is a valid inference. For example:
  
  *John drove to New York, but never got there.*

- If it doesn’t make sense to say “X but not Y”, then Y is part of X’s basic meaning.
  
  *John arrived in New York, but never got there.*

Inferences associated with INGEST

- PTRANS is inferred
- The object ceases to exist in its original form.
- If the object is edible, the actor has less hunger.
- If the object is inedible or toxic, the actor becomes sick.

*Mary ate an apple.*

- The apple is no longer where it originally was and is now inside Mary.
- The apple no longer exists in its original form.
- Mary has less hunger.

Story Understanding

*John was hungry. He went into Goldstein’s and ordered a pastrami sandwich. It was served to him quickly. He left the waitress a big tip.*

- What is Goldstein’s?
- What did John eat?
- Who made the sandwich?
- Who took John’s order?
- Who served the sandwich?
- Why did John leave a big tip?
Scripts

- A script is a stereotypical sequence of events associated with a common activity.
- A script is a conceptual knowledge structure that was proposed as a means of memory organization.
- People usually omit many events in the sequence when telling a story.
- Scripts are essential to story understanding as a mechanism for filling in details that were not explicitly mentioned but would be commonly inferred.

Using the script to make inferences

John was hungry. He went into Goldstein’s and ordered a pastrami sandwich. It was served to him quickly. He left the waitress a big tip.

1. go to a restaurant → John went into Goldstein's.
2. be seated → John sat down.
3. read menu → John read a menu.
4. order food → John ordered a pastrami sandwich.
5. served food → The sandwich was served to John.
6. eat food → John ate the sandwich.
7. pay for meal → John paid and left a tip.
8. leave the restaurant → John left Goldstein's.

A simple restaurant script

$RESTAURANT
1. go to a restaurant
2. be seated
3. read menu
4. order food
5. served food
6. eat food
7. pay for meal
8. leave the restaurant

Each individual event is called a scene. A scene may have its own script.

Script Tracks

A script may have several tracks representing different variations of the situation.

For example, the restaurant script may have tracks for:

Cafeteria: seat yourself, no waiter or waitress, no tipping

Fast food: seat yourself, no waiter or waitress, no tipping, pay after ordering but before eating, may eat inside or outside the restaurant

Fine dining: several waiters and waitresses, wine list, additional courses
A Subway Script

$SUBWAY
1. enter subway station
2. go to turnstile
3. put token in turnstile
4. go through turnstile
5. go to train platform
6. wait for train
7. enter train
8. find a seat
9. exit train at destination

Script Roles

Each script has a cast of characters specified by roles.

For example, $SUBWAY has the roles:

- &PATGRP (patron) group of subway riders
- &CASHIER cashier
- &CONDUCTOR conductor
- &DRIVER person driving the train
- &SUBORG subway organization

Props

Each script has a set of associated objects called props.

For example, $SUBWAY has the props:

- &TOKEN subway token
- &FARE money paid for a token
- &TURNSTILE turnstile
- &PLATSEAT seat on a platform
- &TRAIN the train
- &TRAIN_CAR a car of the train
- &CAR_SEAT a seat on a car
- &STRAP strap for rider to grasp
- &ENTERGATE the gate from the origin to the platform
- &EXITGATE the gate from the platform leading to the destination

Settings

Each script has a list of places where the events take place, called settings.

For example, $SUBWAY has three main settings:

- the origin station
- inside the train
- the destination station

Together, the roles, props, and settings make up the script variables. These variables are instantiated with information from the story, or default values, when the script is applied.
**Question Answering**

John went to a restaurant. He ordered a hot dog. The waiter said they didn’t have any. He asked for a hamburger. When the hamburger came it was burnt. He left the restaurant.

Q1: Did John sit down at the restaurant?  
A1: Probably.

Q2: Did John order a hot dog?  
A2: Yes.

Q3: Did John eat a hot dog?  
A3: No, the waiter told John the management was unable to give it to him.

Q4: What did the waiter serve John?  
A4: The waiter served John a hamburger.

Q5: Why didn’t John eat the hamburger?  
A5: Because the hamburger was overdone.

Q6: Did John pay the check?  
A6: No, John was angry because the hamburger was overdone so he left the restaurant.

**Plans and Goals**

Many stories require knowledge about plans and goals in order to understand how one sentence relates to another.

John needed money for a down payment on a house.  
He called his sister.

John wanted to become a foreman.  
He went to get some arsenic.

John was hungry.  
He took out the yellow pages.  
He took out some ground beef.  
He took out Popular Mechanics. (*)

**Summary**

- The work on conceptual representations and knowledge structures at Yale was revolutionary because they demonstrated that a computer could exhibit deep understanding and make complex inferences.

- However, these systems were extremely knowledge-intensive! BORIS (Dyer 1982) integrated all 22 types of knowledge structures that they developed into a single system. It took 2.5 years to push 2 stories through BORIS, with about 8 people working on the system.

- The resulting sentiment was that conceptual knowledge structures show great promise for deep understanding, but they won’t be practical until we conquer the knowledge-engineering bottleneck.