INFORMATION MAPPING (R. Horn)

Paper metaphors for hypertext

- library card catalogues
- footnotes
- cross-reference
- sticky notes
- commentaries
- indexes
- quotes
- anthologies

Computer metaphors for hypertext

- linked note cards
- popup notes
- linked screens or windows
- stretch text and outlines
- semantic nets
- branching stories
- relational databases
- simulations

Hypertext Links

- *system-supplied*
  - command and control pathways
  - table of contents
  - history tracking
  - automated profiling
- *user-created*
  - detours and shortcuts
  - notes, commentary, reminders
  - analogical links
  - new text
  - links to other knowledge bases
- *author-created*
  - links to prerequisite knowledge
  - hierarchical classification
  - chronological structures

Kinds of links

hierarchical	building a tree
keyword	building an array
referential	building a pointer list
cluster	building a struct

Wayfinding in cyberspace (these don't work very well)

show all connections
go back to the beginning
show history of behavior

Node sizes

one sentence
text of arbitrary size (article, monograph)
index card size
screen size
scroll of any length
variable record sizing
variable size, precisely and flexibly chunked

Information types

structure
concept
procedure
process
classification
principle
fact

Information Blocks

chunking	small, manageable hunks (blocks, maps)
relevance	one main point per chunk, based on purpose or function to reader
consistency	similar words, labels, formats, organization
labeling	label every chunk based on specific criteria

Common types of information blocks

analogy
block diagram
checklist
classification table
classification tree
comment
cycle chart
decision table
definition
description
diagram
example
fact
flow chart
flow diagram
formula
input-procedure-output
non-example
notation
objectives
outlines
parts-function table
parts table
prerequisite
principle
procedure table
purpose
rule
stage
synonym
theorem
when to use
worksheet

Types of hypertrail, path

- prerequisite
- classification
- chronological
    - sequence of events
    - storyline
    - natural development
- geographic
- project
- structural
- decision
- definition
- example

How readers behave

- novices stop reading too soon
- novices are mislead by superficial features
- novices rarely seek non-linear information
- readers construct a hierarchical mental representation
- readers remember the top level of information better
- readers depend on repetition of keywords

Reading cues

- hierarchical text organization
- explicit transitions
- sequence signals
- contrast and similarity cues
- pronouns as cohesiveness cues
- metaphors
- content schemas

Document titles

- just right: not too general, too specific, too long, too short
- common language for the intended audience
- itemize all possible readers and use lowest common denominator
- no cuteness or silliness
- no vague, mislabeled topic headers
- same words in contents, titles, pages, and references

FORMAL KNOWLEDGE

A **conceptual model** consists of

discrete objects, presumed to exist: the *Universe of Discourse*
interrelations between objects
functions: compound names for objects and for unnamed objects
relations: truth statements about objects

No matter how the world is conceptualized, there are other conceptualizations that are just as useful.

Declarative Style

An **knowledge-based program** consists of

a set of objects
a set of functions (names for compound objects)
a set of relations (facts)
a set of permissible transformations

State Space

The collection of facts (the database) at one given time defines the **state** of the world.
All possible state configurations define the **state space**.
To move from one state to another, apply a permitted **transformation rule**.
The state space and the moves between states from a **graph**.
Algorithms explore/search the state space.
Programmers control the search.

Knowledge Representation Labels

Constants:

names of specific objects:	John, Tuesday, My-Phone-Number
names of specific functions:	House-of[x], Phone-of[x], Truth-of[p]
names of specific relations:	Likes[Mary, Tom], Phone-Number[Tom, x]

Variables:

refer to sets/classes/domains of objects
always scoped/introduced by a quantifier

Knowledge Representation Atoms

Named objects	(object constants)
Indirect/compound named objects	(functions)
Relations between objects	(facts)

Logical connectives (and, or, not, if, iff) connect atoms. They cannot be used inside atoms.

```
yes:    eyes-of[John] AND hair-of[John}
no:     (eyes-of AND hair-of)[John]
no:     hair-of[John AND Mary]
yes:    hair-of[John] AND hair-of[Mary]
```

Example of a RELATIONAL KNOWLEDGE-BASE

Part of a knowledge-base about family relationships.

Vocabulary:

```
        (father X Y)
        (mother X Y)
        (male Y)
        (female Y)
        (parent X Y)
        (sibling X Y)
        (brother X Y)
        (sister X Y)
        (uncle X Y)
        (aunt X Y)
        (gfather X Y)
        (gmother X Y)
        (ancestor X Y)
        (cousin X Y)
```

Knowledge Base:

```
        (if (father A B) (parent A B))
        (if (mother A B) (parent A B))
        (if (and (parent A C) (parent A B) (not (= B C))) (sibling B C))
        (if (and (sibling A B) (male A)) (brother A B))
        (if (and (sibling A B) (female A)) (sister A B))
        (if (and (parent B C) (brother A B)) (uncle  A C))
        (if (and (parent B C) (sister A B)) (aunt  A C))
        (if (and (parent B C) (father A B)) (gfather  A C))
        (if (and (parent B C) (mother A B)) (gmother  A C))
        (if (parent A B) (ancestor A B))
        (if (and (parent A B) (ancestor B C)) (ancestor  A C))
        (if (and (parent A C) (parent B D) (sibling A B)) (cousin C D))
        (if (father A B) (male A))
        (if (mother A B) (female A))
```

Facts:

```
        (father arthur bertram)
        (father arthur bailey)
        (father bertram cornish)
        (father bertram carey)
        (mother beatrice cornish)
        (mother beatrice carey)
```

```
(father bailey carleton)
(father bailey cassandra)
(mother bessie carleton)
(mother bessie cassandra)
(male cornish)
(male carey)
(male carleton)
(female cassandra)
```

Example questions:

```
(gfather arthur ?)
(cousin ? cassandra)
```

Technical Difficulties in Modeling and Knowledge Representaiton

(using blocks world in LISP as an example)

1. What is important to describe?
   Build little theories of little worlds.
   `(Block A)    (OnTable A)    (Hand Empty)`

2. How should descriptions be partitioned?
   Functions or Relations, special or general objects?
   `(OnTable A)    (On A Table)    (not (OnTable Table))`

3. How do we talk about groups and classes of objects?
   Quantification and abstraction
   `(All (x) (Block x))`

4. How do we address things with no names?
   Functions as indirect, compound names.
   `(House-of John)`

5. How do we handle things with more than one name?
   unique name hypothesis, unification
   `(Uncle John) = (Brother (Father John)) = Bob`

6. How do we make general rules which define the structure of relations?
   quantification
   `(All (x) (iff (Uncle x) (Brother (Father x))))`

7. How are typing and filters on domains represented?
   predicates in conjunction
   `(All (x) (and (Person x) (Father x y)))`

8. How do we join more than one fact?
   conjunction
   `(and (F x) (G x))`

9. How do we compute with logic?
   inference as natural deduction and as resolution
   (if (and (P x) (if (P x) (Q x))) (Q x))

10. How do we compute with quantifiers and classes of objects?
    implicit universal quantification, Skolemization
    (Exist (x) (P x)) ==> (P (Sk-1 x))

11. What is the difference between a fact and a query?
    query combination rules

    A. conjunction with negated query
       (and (P x) (not (Q ?)))

    B. Skolemization of query variables
       (Q ?) ==> (Q Sk-1)

    C. Facts imply Query
       (if (P x) (Q ?))

    D. The answer predicate
       (if (P x) (Answer x))

12. What kinds of rules do we need for query answering?

    A. definitions
       (iff (P x) (Q (R x)))

    B. mathematical structures (symmetry, transitivity, etc)
       (if (and (if (P x) (Q x)) (if (Q x) (R x))) (R x))

    C. permissible state transformations
       (Pick-up x) = (Assert (not (onTable x)))

13. How can we control the inference/search procedure?

    A. Pre and Post conditions
    B. Compound queries
    C. Searching databases of rules and facts

14. How do we steer the resolution process?

    A. set of support
    B. ordered resoultion
    C. static vs dynamic approaches (compiled vs run-time)
    D. lookahead, cheapest first, dependency directed search

15. How do we express meta-level reasoning (rules about rules)
    measure the savings vs brute force