Molecular Expert Systems

Rob Kitto

November 1998

Topics

- **DNA Computing**
- **Expert Systems**
- **Molecular Expert Systems**

DNA Computing

A computer is a device for storing and processing data.

Basic idea:

- **Encode information**
- **Perform syntactic manipulations**
- **Decode output**

This definition is platform independent. It could apply to:

- **An abacus**
- **An electronic computer**
- **A DNA-based computer**
- **A Quantum computer**
- **???**

Basics of DNA Computing

- **4 natural bases, represented by C,G,A,T**
- **C and G are complementary, as are A and T**
- **Sequences of bases can be joined to form arbitrarily long strands, or *oligonucleotides***
- **Oligos have a polarity: 5’ to 3’ or 3’ to 5’**
- **2 oligos *should* form a double strand iff:**
  - **they have opposite polarity**
  - **corresponding bases on opposite strands are complementary**
- **For example, 5’CGA3’ and 3’GCT5’ should bind, but 5’CGA3’ should not bind with 5’GCT3’ or with 3’CGT5’**

...DNA Computing

Reality:

- **Molecules of DNA are TINY**
- **They are awkward to work with**
- **They don't always bind as they should**
- **It is difficult to decode the output of a DNA computation**

More Reality:

- **People used to 'program' electronic computers by manually rearranging the wiring**
- **Input and output was done with punchcards**
- **The ENIAC had 18 000 vaccum tubes, and could only perform hundreds of multiplications per minute.**

Expert Systems

- **What is an Expert System?**
- **History and Background**
- **Sample use**
- **Some Technical Details**
- **Example**

What is an Expert System?

Basically, the purpose of an expert system is to be able to do automated reasoning within a small, well-defined framework or domain.

There are several types of expert systems. For example, Belief Networks or Truth Maintenance Systems.

We will focus on ones that have a knowledge base (KB) of clauses and use resolution as an inference engine.

History / Background

Different Types of Logic problems:

- **Satisfiability**
- **Formal Deduction**
- **Resolution with clauses**

The first two types are computationally very slow. Resolution, especially under the correct conditions, can be quite fast.

Dates

- **1965: Robinson introduces resolution rule from work in AI and automatic theorem proving**

- **1969: ‘Dendral’ developed to infer the structure of molecules. This is arguably the first expert system.**

- **1972: Kowalski and Colmerauer introduce idea of using logic as a programming language (Prolog).**

Currently, many expert systems use a resolution style inference procedure (frequently implemented in Prolog).

Uses

Medical Diagnosis:

KB stores rules relating symptoms and diseases. For example:

R_n: loss of memory, discombobulation, >50 years → Alzheimer's disease

The KB would contain hundreds of rules.

The inference engine would be given a list of symptoms, and would return a list of possible diseases.

A doctor could then consult this list and decide what tests to run.

Some Technical Details

Definitions:

A *Literal* is an atom or the negation of an atom.

A *Clause* is a disjunction of literals

A *Horn Clause* is a clause with at most one positive literal
(eg: $A \vee \sim B_1 \vee \ldots \vee \sim B_n$)

Generally, we use the equivalent form:

$$A \leftarrow B_1 B_2 \ldots B_n$$

for a Horn Clause.

All the information in our KB will be stored in Horn clauses.

Resolution

The inference engine will be repeated applications of the *Resolution* rule:

$$\frac{C \cup \{L\},\ D \cup \{\sim L\}}{C \cup D}$$

We say we are *resolving clauses over L*, and $C \cup D$ is the *resolvent*.

The Davis-Putnam procedure (DPP) is an algorithm for computing resolvents from a set of clauses.

The DPP has been shown to be both sound and complete.

...Resolution

Generally in an expert system, there is a conclusion we want to (dis)prove. I will refer to this as the *hypothesis*.

The basic method is:

- **-Add negation of hypothesis to KB**
- **-Apply resolution until either {} or Ø is derived.**
- **-if {} is derived, the KB supports the hypothesis**
- **-if Ø is derived, it does not**

Note: it may be convenient to think of resolution in terms of backward chaining. The following example should demonstrate why the two are essentially equivalent.

Example

A - Engine works irregularly, or does not reach full power.
B - Engine is not overheated
C - Fuel is reaching carburetor
K - Possible damage of carburetor
L - Carburetor is operating incorrectly
M - Impurity of fuel nozzle, water in the carburetor or fuel pipe problem

Facts: A,B,C
Rules:
- R1: $C \wedge A \rightarrow K$
- R2: $K \wedge B \rightarrow L$
- R3: $L \rightarrow M$

Hypothesis: M.

Molecular Expert Systems

Jan Mulawka, Piotr Weglenski and Piotr Borsuk have devised a method which can solve the previous example using simple operations on strands of DNA.

To explain this method, we'll look at:

- **How information is encoded**
- **How the inference process works**
- **The previous example from this perspective**
- **Their experimental results**
- **Some commentary on their method**

The Knowledge Base

Rules are defined as having the following form:

If (premise 1 and... premise n) Then k

This is essentially the Horn clause:

$$k \leftarrow p_1 \ldots p_n$$

Facts are rules with no premises.

All propositions (premises, facts, hypothesis) will be represented or encoded in DNA according to an encoding scheme E.

The Encoding Scheme E

- There are various ways propositions can be encoded in DNA... the details are not important here.

- Certain properties in E can reduce the chance of unwanted DNA binding.

- Rule If (a and b) Then c would be encoded as: $\overline{ab}c$

- Fact a would be encoded: a

- Hypothesis h would be: $\overline{h}$

Inference Engine

- The natural attraction of complementary DNA strands serves as the inference engine.

- Success criterion: A complete double strand, with both ends blunt.

- How to check for success...

- Example

Experimental Results

- **The example was successfully encoded, and a complete double strand formed (proving the hypothesis).**

- **Several versions of the experiment were also performed with one or more oligo (representing a rule or fact) not included in the mix. In these experiments, a double strand was *not* formed. This is good, because the KB should need all the facts and rules to support the hypothesis.**

Commentary

Time and space complexity should both be relatively low.

A *false positive* occurs when the experiment proves the hypothesis, but it is not logically entailed by the KB. These should be easy to detect.

A *false negative* occurs when the experiment does not prove the hypothesis, but it is logically entailed by the KB. These present more of a problem.

Only a limited backward chaining algorithm can be performed with this method.

Conclusion

Pros:

- **Lab implementation**
- **Favourable complexity**
- **Uses ‘self-assembly’ of molecules**

Cons:

- **Limited backward chaining**
- **Strictly propositional**

ERROR: undefined
OFFENDING COMMAND: ^D

STACK: