## AI Course Fall 2015:
## Exercise 3: Still Constraint Satisfaction, Propositional/Predicate Logics

Hand-out: Nov. 26th 2014
Discussion: Dec. 2rd 2014

### Task 7: Constraint Satisfaction: Cryptoarithmetic Puzzles

Solve the following cryptoarithmetic puzzle using backtracking search with constraint propagation (forward checking and arc-consistency checking).

$$\begin{array}{r} S\ E\ N\ D \\ M\ O\ R\ E \\ \hline M\ O\ N\ E\ Y \end{array}$$

allDiff{S, E, N, D, M, O, R, Y}

$M \in \{1,2,3,4,5,6,7,8,9\}$
$S \in \{1,2,3,4,5,6,7,8,9\}$

$E, O, N, D, R, Y \in \{0, 1,2,3,4,5,6,7,8,9\}$

D + E = Y +X1*10
N + R + X1 = E * 10+ X2
E + O + X2  = N * 10+ X3
S + M + X3 = O * 10+ X4
M = X4

→ M=1; S + 1 = 10 * 1 + X3 → S=9 ; as O cannot be 1 →O=0 and X3=0
→ E + 0 + x2 = N *10; E not equal to N → x2 must be 1 → E+1=N

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| S | | | | | | | | | | |
| E | | | | | | | | | | |
| N | | | | | | | | | | |
| D | | | | | | | | | | |
| M | | | | | | | | | | |
| O | | | | | | | | | | |
| R | | | | | | | | | | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Y | ■ | ■ | | | | | | | | ■ |
| X1 | | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| X2 | ■ | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| X3 | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| X4 | ■ | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |

So far so good with the preprocessing.
The variable with the lowest number of values is now X1:
Lets set x1 = 0 →
D + E = Y
N + R  = E + 10
E+1=N → E+1+ R = E+10 →R = 9 → but S=9 so there is an inconsistency → take back assignment and back propagate. There is another value for X1:

X1=1
N + R + 1  = E + 10
E+1=N → E+1+ R + 1 = E+10  → R = 8 (because of E+1=N, E cannot be 7)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| S | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | |
| E | ■ | ■ | | | | | | ■ | ■ | ■ |
| N | ■ | ■ | ■ | | | | | | ■ | ■ |
| D | ■ | ■ | | | | | | | ■ | ■ |
| M | ■ | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| O | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| R | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | | ■ |
| Y | ■ | ■ | | | | | | | ■ | ■ |
| X1 | ■ | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| X2 | ■ | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| X3 | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |
| X4 | ■ | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ |

So, lets test E (more constraints than N)

Add assignment E=2 and test:

D + 2 = Y + 10→ D must be 8 or 9, these numbers are not available anymore … there is no consistent combination at D when E=2  and thus when N=3 → both values have to be deleted.
N = 3
3 + 8 + 1 = 2 + 10

Backtrack

Add assignment E =3 and test:

D + 3 = Y + 10 → D would be 7 and Y would be 0 → but Y cannot be 0 as O is already 0 → E cannot be 3 and N cannot be 4

Backtrack:

Add assignment E = 4 and test:

D + 4 = Y + 10 → D could be 6 or 7, if D would be 6 → Y would be 0, no;
If D would be 7 → Y would be 1, no as well.
→ E cannot be 4 (and N cannot be 5)

Backtrack:

Add assignment E = 5

D + 5 = Y + 10 → D could be 5,6,7

Add assignment D = 5 -> Y= 0, no; → backtrack

Add assignment D = 6 → Y=1, no; → backtrack

Add assignment D = 7 → There is a consistent value Y =2

Add assignment N = 6 from E+1=N;
lets look at the table now:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| S | | | | | | | | | | |
| E | | | | | | | | | | |
| N | | | | | | | | | | |
| D | | | | | | | | | | |
| M | | | | | | | | | | |
| O | | | | | | | | | | |
| R | | | | | | | | | | |
| Y | | | | | | | | | | |
| X1 | | | | | | | | | | |
| X2 | | | | | | | | | | |
| X3 | | | | | | | | | | |
| X4 | | | | | | | | | | |

Ok, we are done.

---

## Task 8 Describing a situation using First Order Logics

Invent a set of predicates and object/entity descriptors for describing how your writing table looks like: What type of items populated the surface? Invent a language with different predicates and describe the situation using predicate logic.

What objects are on my desk?

There are books, paper and a computer

What functional relations do I have to consider on my desk – functional relations in the sense that a function points to unique other object? No.

What predicates describe the objects and their relations.

Type information: book(x), paper(y)

Readable(x) – the title of an item is readable; unary predicate, works for books and papers

Visible(y) – the item is not visible, that is even harder than readable

On(x,y) – x is on y

Let's define a little the "physics" / universal laws of my desk:

I can read only book titles and paper titles if it is the top:

$\forall x\ [\neg\exists\ y\ on(y,x)] \wedge [book(x) \vee paper(x)] \Rightarrow readable(x)$

Everything which is readable is also visible:

$\forall x\ readable(x) \Rightarrow visible(x)$

Nothing which is below two items is visible

$\forall x\ \exists y, \exists z\ on(y,x) \wedge on(z,x) \wedge \neg(z=y) \Rightarrow \neg visible(x)$

## Task 9 Truth Tables for Satisfiability Test

Use the Model Checking/Truth Table Method for checking whether the following knowledge base entails the statement: **B**

$A \Rightarrow B$
$(\neg A \wedge \neg K)$ or $(A \wedge K)$
K

| A | B | K | $A \Rightarrow B$ | $(\neg A \wedge \neg K)$ | $(A \wedge K)$ | $(\neg A \wedge \neg K)$ or $(A \wedge K)$ | KB |
|---|---|---|---|---|---|---|---|
| False | False | False | True | True | False | True | False |
| False | False | True | True | False | False | False | False |
| False | True | False | True | True | False | True | False |
| False | True | True | True | False | False | False | False |
| True | False | False | False | False | False | False | False |
| True | False | True | False | False | True | True | False |
| True | True | False | True | False | False | False | False |
| True | True | True | True | False | True | True | True |

B is true in all cases in which the KB is true → B is entailed in the knowledge base.

## **Task 10 CNF (Propositional Logic)**

Transform the following sentences to Conjunctive Normal Form $(\vee\vee)\wedge(\vee\vee)\ldots\wedge(\vee\vee)$

a) $A \Leftrightarrow B$

$A \Leftrightarrow B \quad \equiv (B \Rightarrow A) \wedge (A \Rightarrow B)$

$\equiv (\neg B \vee A) \wedge (\neg A \vee B)$

b) $(A \wedge B) \Leftrightarrow (A \vee B)$

$(A \wedge B) \Leftrightarrow (A \vee B) \quad \equiv ((A \wedge B) \Rightarrow (A \vee B)) \wedge ( (A \vee B) \Rightarrow (A \wedge B))$

$\equiv (\neg (A \wedge B) \vee (A \vee B)) \wedge ( \neg (A \vee B) \vee (A \wedge B))$

$\equiv ([\neg A \vee \neg B] \vee (A \vee B)) \wedge ( [\neg A \wedge \neg B] \vee (A \wedge B))$

$\equiv (\neg A \vee \neg B \vee A \vee B) \wedge ( [(\neg A \wedge \neg B) \vee A] \wedge [(\neg A \wedge \neg B) \vee B])$

$\equiv \text{TRUE} \quad \wedge ( [(\neg A \vee A ) \wedge (\neg B \vee A)] \wedge [(\neg A \vee B ) \wedge (\neg B \vee B])$

$\equiv [\text{TRUE} \wedge (\neg B \vee A) \wedge (\neg A \vee B ) \wedge \text{TRUE})$

$\equiv (\neg B \vee A) \wedge (\neg A \vee B )$

c) $A \wedge (A \Rightarrow B) \Rightarrow B$

| | | |
|---|---|---|
| $A \wedge (A \Rightarrow B) \Rightarrow B$ | $\equiv \neg (A \wedge (A \Rightarrow B)) \vee B$ | 1. Resolve $\Rightarrow$ |
| | $\equiv \neg (A \wedge (\neg A \vee B)) \vee B$ | |
| | $\equiv \neg A \vee \neg (\neg A \vee B) \vee B$ | 2. Bring in the nots |
| | $\equiv \neg A \vee (A \wedge \neg B) \vee B$ | haha, now we have DNF |
| | $\equiv [\neg A \vee (A \wedge \neg B) ] \vee B$ | ok, Distribution Laws again… |
| | $\equiv [(\neg A \vee A) \wedge (\neg A \vee \neg B) ] \vee B$ | |
| | $\equiv [\text{TRUE} \wedge (\neg A \vee \neg B) ] \vee B$ | |
| | $\equiv \neg A \vee \neg B \vee B$ | get rid of the brackets |
| | $\equiv \neg A \vee \text{TRUE}$ | |
| | $\equiv \text{TRUE}$ | |

## Task 11 Resolution Proofs in Propositional Logic

Consider the following logic puzzle (taken from the Ertel-Book)

Three girls practice high jump for their physical education test. The bar is set to 1.20meters. “I bet” says the first girl to the second, “that I will make it over if, and only if you don’t”. If the second girl said the same to the third, who in turn said the same to the first, would it be possible for all three to win their bets?

Formalize the statements in propositional logic, translate it into Conjunctive Normal Form and use the resolution proof to test whether it is possible for all three to win their bets.

Our hypothesis is that they cannot win all three their bets → so, we negate the question and test the knowledge base

Question: $\neg[(A \Leftrightarrow \neg B) \wedge (B \Leftrightarrow \neg C) \wedge (C \Leftrightarrow \neg A)]$

That means we take $[(A \Leftrightarrow \neg B) \wedge (B \Leftrightarrow \neg C) \wedge (C \Leftrightarrow \neg A)]$
(the negated question equals the set of the statements) and test whether there is a contradiction or not.

$A \Leftrightarrow \neg B$
$B \Leftrightarrow \neg C$
$C \Leftrightarrow \neg A$

$(A \Rightarrow \neg B) \wedge (\neg B \Rightarrow A) \equiv (\neg A \vee \neg B) \wedge (A \vee B)$

$(\neg B \vee \neg C) \wedge (B \vee C)$

$(\neg C \vee \neg A) \wedge (C \vee A)$

That means your knowledge base is

1$(\neg A \vee \neg B)$
2$(A \vee B)$
3$(\neg B \vee \neg C)$
4$(B \vee C)$
5$(\neg C \vee \neg A)$
6$(C \vee A)$

So, let’s start resolving

1+6 → 7 $(C \vee \neg B)$

4+7 → 8 C

2+5 → 9 (B ∨¬C)

3+9 → 10 ¬C

8+10 → contradiction

## Task 12 Unification

For each pair of atomic sentences, give the most general unifier, if it exists.

a) P(A,A,B) and P(x,y,z).

using the algorithms given in the lecture:

Unify(P(A,A,B),P(x,y,z),{})

→ Unify((A,A,B),(x,y,z),unify(P,P, {}))
unify(P,P, {}) returns {}

Unify((A,B),( y,z),unify(A, x, {}))
unify(A, x, {}) returns {x|A}
unify((B),(z), unify(A,y, {x|A})
unify(A,y, {x|A}) returns {x|A, y|A}
unify(B,z, {x|A, y|A}=
returns {x|A, y|A, B|z}

b) Q(y,G(A,B)) and Q(G(x,x),y).

→ Unify(Q(y,G(A,B)), Q(G(x,x),y), {})
Q=Q ✓
→ Unify((G(A,B)), y), unify(y,G(x,x), {})
unify-var(y, G(x,x), {}) add {y|G(x,x)}
→ Unify(G(A,B), y), {y|G(x,x)})
unify-var(y, G(A,B), {}) → there is already an assignment of y|G(A,B) there →

Unify (G(x,x), G(A,B),{ y|G(A,B)}
Failure as x|A and then x|B fails as x cannot be connected to both A and B

c) Older(Father(y),y) and Older(Father(x),Jerry)

Unify(Older(Father(y),y), Older(Father(x),Jerry), {}

→ Unify((Father(y),y), (Father(x),Jerry), {}

→→ unify(y,Jerry, unify(Father(y), Father(x), {})

→→ unify(y,Jerry, {x|y})

→→{y|Jerry, x|y} = {y|Jerry, x|Jerry}

d) Knows(Father(x),x) and Knows(y,y)

unify(Knows(Father(x),x),Knows(y,y), {})

→ unify((Father(x),x), (y,y), {}) → unify(x, y,  unify(Father(x), y, {}))
→unfiy-var(y, Father(x), {}) → {y|Father(x)}

→ unify-var(x,y) – y is Father(x) → unify-var(x, Father(x)) → occur-check!

(small letters are variables, capitalized words refer to terminals, functions or predicates)