Assignment 2. Due: 02/09/15

4. Construct a controlled grammar for solving the generalized Missionaries and Cannibals Problem and show the derivation:

(a) in case of 4 missionaries and 4 cannibals assuming that boat can carry 3 people.

(b) in case of *n* missionaries and *n* cannibals (make your own assumption about the boat).

5. Construct a controlled grammar for solving the Tower of Hanoi Problem and show the derivation in case of 4 pivots and *n* disks. Can your grammar generate a solution in a lesser number of steps than the grammar employing only 3 pivots (which was shown in class)?

Hierarchy of Languages

A set of dynamic subsystems might be represented as a hierarchy of formal languages where each "sentence" (a group of "words" or symbols) of the lower level language corresponds to the "word" of the higher level one. This is a routine procedure in our native language. For example, the phrase "**A man who teaches students**" creates a hierarchy of languages. A lower level language is a native language without the word "professor." The symbols of this language are all the English words (except "professor"). A higher level language might be the same language with one extra word **"A-man-who-teaches-students"**. Instead, we can use the word **"professor"** which is simply a short designation of this long word.

Following a linguistic approach each subsystem could be represented as a string of symbols with parameters: $\boldsymbol{a}(x_1)\boldsymbol{a}(x_2)...\boldsymbol{a}(x_n)$, where the values of the parameters incorporate the semantics of the problem domain or lower-level subsystems.

See also:
Chapter 8 of the textbook.

Optional Reading:

- Stilman, B., Discovering the Discovery of the Hierarchy of Formal Languages, *Int. J. of Machine Learning and Cybernetics*, Springer, 10.1007/s13042-012-0146-0, 25 p, 2012.
- Stilman, B., Discovering the Discovery of the No-Search Approach, *Int. J. of Machine Learning and Cybernetics*, Springer, (DOI) 10.1007/s13042-012-0127-3, 27 p., 2012.
- Stilman, B., Discovering the Discovery of Linguistic Geometry, *Int. J. of Machine Learning and Cybernetics*, Springer, (DOI) 10.1007/s13042-012-0114-8, 20 p., 2012.
- Stilman, B., Visual Reasoning for Discoveries, *Int. J. of Machine Learning and Cybernetics*, Springer, (DOI): 10.1007/s13042-013-0189-x, 23 p., 2013.

Formal Grammars

Let us parse the following sentence:

The girl walks gracefully.
noun phrase   verb phrase

<sentence>
<noun phrase> <verb phrase>.
<adjective> <noun> <verb phrase>.
The <noun> <verb phrase>.
The girl <verb phrase>.
The girl walks <adverb>.
The girl walks gracefully.

We can introduce the following rules (productions):

1. <sentence> —> <noun phrase> <verb phrase>
2. <noun phrase> —> <adjective> <noun>
3. <verb phrase> —> <verb> <adverb>
4. <adjective> —> The
5. <noun>—> girl
6. <verb> —> walks
7. <adverb> —> gracefully

where symbol “**—>**“ means “**can be replaced**”.

Instead, we can use capital letters:

1. S —> A B
2. A —> C N
3. B —> V D
4. C —> The
5. N —> girl
6. V —> walks
7. D —> gracefully

Instead, we can use capital letters:

1. S —> A B
2. A —> C N
3. B —> V D
4. C —> The
5. N —> girl
6. V —> walks
7. D —> gracefully

Now we can represent parsing as follows:

S $\xrightarrow{1}$ A B
$\xrightarrow{2}$ C N B
$\xrightarrow{4}$ The N B
$\xrightarrow{5}$ The girl B
$\xrightarrow{3}$ The girl V D
$\xrightarrow{6}$ The girl walks D
$\xrightarrow{7}$ The girl walks gracefully

General formal grammar is the following 4-tuple:

$$G = (V_T, V_N, P, S)$$

In our case:

V_T ={The, girl, walks, gracefully}:	terminal symbols
V_N ={A, B, C, D, N, V}:	nonterminal symbols
P:	7 productions
S:	start symbol

Controlled Grammars

Informal Definition

Label	Condition	Kernel	F_T	F_F
l	Q(, ,)	A(, ,) —> B(, ,)	L_T	L_F

Parameters (variables and functions) are shown in parenthesis.

If condition Q is true, production with label *l* is applied and we go to the production with label from L_T.

If Q is not true, production l does not apply and we go to production from L_F.

Values of parameters are changed when we apply productions.

Controlled Grammars

A ***controlled grammar*** **G** is the following eight-tuple:

$$\mathbf{G} = (\boldsymbol{V_T, V_N, V_{PR}, E, H, Parm, L, R}),$$

where

$\boldsymbol{V_T}$ is the alphabet of *terminal symbols*;

$\boldsymbol{V_N}$ is the alphabet of *nonterminal symbols*, S (from V_N) is the start symbol;

$\boldsymbol{V_{PR}}$ is the alphabet of the *first order predicate calculus PR*:

V_{PR} = *Truth* U*Con* U*Var* U*Func* U*Pred* U{symbols of logical operations},

where

Truth are truth symbols T and F (these are reserved symbols);

Con are constant symbols;

Var are variable symbols;

Func are functional symbols (*Func* = *Fcon* U*Fvar*). Functions have an attached non-negative integer referred to the *arity* indicating the number of elements of the domain mapped onto each element of the range. A term is either a constant, variable or function expression.

A *function expression* is given by a functional symbol of arity k, followed by k terms, $t_1, t_2, \ldots, t_k$, enclosed in parentheses and separated by commas;

Pred are predicate symbols. Predicates have an associated positive integer referred to as *arity* or “argument number” for the predicate. Predicates with the same name but different arities are considered distinct. An *atom* is a predicate constant of arity n, followed by n terms, $t_1, t_2, \ldots, t_n$, enclosed in parentheses and separated by commas. The truth values, T and F, are also atoms. *Well-formed formulas* (or WFF) are atoms and combinations of atoms using logical operations;

H is an *interpretation* of *PR* calculus on the set *E*,

Parm is a mapping from $V_T \mathrm{U} V_N$ in 2^{Var} matching with each symbol of the alphabet $V_T \mathrm{U} V_N$ a set of formal parameters, with $Parm(S)=Var$;

L is a finite set called the set of *labels*;

R is a finite set of *productions*, i.e., a finite set of the following seven-tuples:

$$(\boldsymbol{l, Q, A \longrightarrow B, \pi_k, \pi_n, F_T, F_F}).$$

Controlled Grammars (continued)

$$(\boldsymbol{l, Q, A \longrightarrow B, \pi_k, \pi_n, F_T, F_F}).$$

Here

_l_ (from *L*) is the label of a production; the labels of different productions are different, and subsequently sets of labels will be made identical to the sets of productions labeled by them;

Q is a WFF of the predicate calculus *PR* , the *condition* of applicability of productions; *Q* contains only variables from *Var* which belong to *Parm*(*A*);

A—>B is an expression called the *kernel of production*, where

A is from V_N;

B is from $(V_T \cup V_N)^*$ is a string in the alphabet of the grammar G;

$\boldsymbol{\pi_k}$ is a sequence of functional formulas corresponding to all formal parameters of each entry of symbols from $V_T \cup V_N$ into the strings *A* and *B* (*kernel actual parameters*);

$\boldsymbol{\pi_n}$ is a sequence of functional formulas corresponding to all formal parameters of each functional symbol from *Fvar* (*non-kernel actual parameters*);

$\boldsymbol{F_T}$ is a subset of *L* of labels of the productions permitted on the next step of derivation if *Q*=T (“true”); it is called a *permissible set in case of success*;

$\boldsymbol{F_F}$ is a subset of *L* of labels of the productions permitted on the next step of derivation if *Q*=F (“false”); it is called a *permissible set in case of failure*.

Structure of a typical controlled grammar

L	***Q***	**Kernel, π_k**	π_n	F_T	F_F
l_i	Q_i	$A(\,,\,,\,) \longrightarrow a(\,,\,,\,)B(\,,\,,)$			
	$V_T = \ldots$	$V_N = \ldots$	$V_{PR} = \ldots$		
	***E* is ...**	***Parm*: ...**			

The Missionaries and Cannibals Problem

Three missionaries and **three** cannibals find themselves on one side of a river. They have agreed that they would all like to get to the other side. But the missionaries are not sure what else the cannibals have agreed to. So the missionaries want to manage the trip across the river in such a way that the number of missionaries on either side of the river is never less than the number of cannibals who are on the same side. The only boat available holds only two people at a time. How can everyone get across the river without the missionaries risking being eaten?

Missionaries and Cannibals

c is the number of cannibals on side B of the river
m is the number of missionaries on side B of the river

L	Q	Kernel	F_T	F_F
1	Q_1	$S(c, m) \longrightarrow A(c + 1, m + 1)$	2	$\varnothing$
2	Q_2	$A(c, m) \longrightarrow p(c, m)A(f_1(c, m), f_2(m))$	2	3
3	Q_3	$A(c, m) \longrightarrow p(c, m)$	$\varnothing$	$\varnothing$

All steps of this grammar take place when the boat is on side B of the river.

Pred = { Q_1, Q_2, Q_3}

Q_1 = **T**

$Q_2(c, m)$ = **T** if $c > 0$ or $m > 0$; $Q_2(c, m)$ = **F** if $((c=3) \wedge (m = 3))$ or $(c > m) \vee ((3 - c) > (3 - m))$

$Q_3(c, m)$ = **T** if $((c = 3) \wedge (m = 3))$

Var = {c, m}

Fcon = $\{f_1, f_2\}$

$f_1(c, m)$ is defined as follows:

```
If (c = 1) ∧ (m = 1)
   f1(c, m) = c+ 2
   else
   if (c ≠ 1) ∧ ( m ≠ 3)
      f1(c, m) = c – 1
      else
      f1(c, m) = c + 1.
```

$f_2(m)$ is defined as follows:

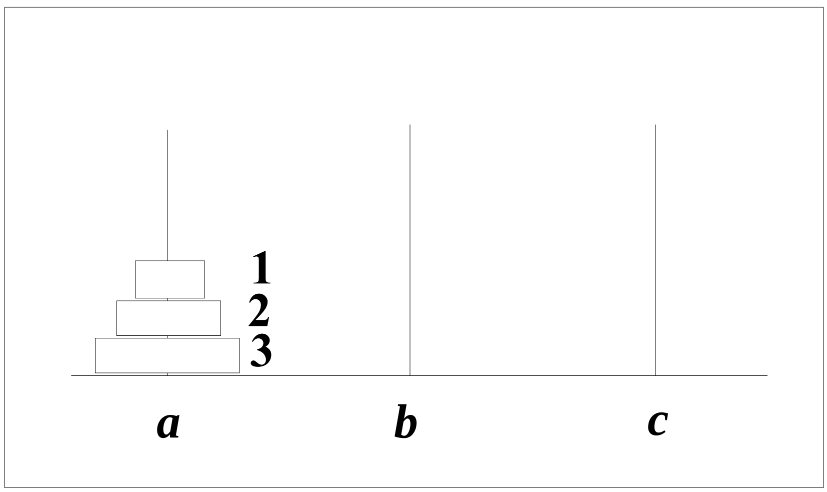
```
If m = 1
   f2(m) = m – 1
   else
   if m = 0
      f2(m) = m + 2
      else
      if m = 3
         f2(m) = m
         else
         f2(m) = m  + 1
```

At the beginning of derivation: c = 0, m = 0

$$
\begin{aligned}
\boldsymbol{S}(0, 0) &\overset{1}{\Rightarrow} \boldsymbol{A}(1, 1) \\
&\overset{2}{\Rightarrow} \boldsymbol{p}(1, 1)\boldsymbol{A}(3, 0) \\
&\overset{2}{\Rightarrow} \boldsymbol{p}(1, 1)\boldsymbol{p}(3, 0)\boldsymbol{A}(2, 2) \\
&\overset{2}{\Rightarrow} \boldsymbol{p}(1, 1)\boldsymbol{p}(3, 0)\boldsymbol{p}(2, 2)\boldsymbol{A}(1, 3) \\
&\overset{2}{\Rightarrow} \boldsymbol{p}(1, 1)\boldsymbol{p}(3, 0)\boldsymbol{p}(2, 2)\boldsymbol{p}(1, 3)\boldsymbol{A}(2, 3) \\
&\overset{2}{\Rightarrow} \boldsymbol{p}(1, 1)\boldsymbol{p}(3, 0)\boldsymbol{p}(2, 2)\boldsymbol{p}(1, 3)\boldsymbol{p}(2, 3)\boldsymbol{A}(3, 3) \\
&\overset{3}{\Rightarrow} \boldsymbol{p}(1, 1)\boldsymbol{p}(3, 0)\boldsymbol{p}(2, 2)\boldsymbol{p}(1, 3)\boldsymbol{p}(2, 3)\boldsymbol{p}(3, 3)
\end{aligned}
$$

The Tower of Hanoi Problem

The problem is as follows. There are three pivots ***a***, ***b***, and ***c***. On the first one there is a set of *n* disks, each of different radius. The task is to move all the disks to the pivot ***c*** moving only one disk at a time. In addition, at no time during the process may a disk be placed on top of a smaller disk. The pivot ***c*** can, of course, be used as a temporary resting place for the disks.



Let us designate an elementary step of moving disk number i from the pivot x to the pivot y as ***p***(i, x, y), a terminal symbol with parameters. Thus, a solution of the Tower of Hanoi Problem can be represented as the following string of symbols with parameters:

$$\boldsymbol{p}(i_1, x_1, y_1)\boldsymbol{p}(i_2, x_2, y_2)\ldots\boldsymbol{p}(i_m, x_m, y_m).$$

This is the string of the language of all possible sequences of moves.

We will construct a specific controlled grammar and apply this grammar for generating a solution for the case of three disks: n =3, x = ***a***, y = ***c***. It means that the values of parameters for the start symbol ***S*** are ***S***(3, ***a***, ***c***).

Controlled grammar generating solutions to the Tower of Hanoi Problem

L	*Q*	Kernel, π_k	π_n	F_T	F_F
1	Q_1	$S(\text{n, x, y}) \to A(\text{n, x, y})$		2	ø
2	Q_2	$A(\text{n, x, y}) \to A(f_1(\text{n}), \text{x}, f_2(\text{x, y}))$ $p(\text{n, x, y})$ $A(f_1(\text{n}), f_2(\text{x, y}), \text{y})$		2	3
3	Q_3	$A(\text{n, x, y}) \to p(\text{n, x, y})$		2	ø

Here $V_T = \{p\}$

$V_N = \{S, A\}$

V_{PR}

Pred = $\{Q_1, Q_2, Q_3\}$,

$Q_1 = T$

$Q_2(\text{n}) = T$, if n > 1; $Q_2(\text{n}) = F$, if n = 1.

$Q_3(\text{n}) = T$, if n = 1; $Q_3(\text{n}) = F$, if n > 1.

Var = {n, x, y}

F = *Fcon* ∪ *Fvar*,

Fcon = $\{f_1, f_2\}$

$f_1(\text{n}) = \text{n}-1$, n = 2, 3,...

$f_2(\text{x, y})$ yields the value from $\{a, b, c\}$ – {x, y}, where values of x, y are from $\{a, b, c\}$

Fvar = $\{3, a, c\}$

$E = \mathbf{Z}_+ \cup \{a, b, c\}$

Parm: $S \to Var$, $A \to Var$, $p \to Var$

$L = \{1, 2, 3\}$

At the beginning of derivation: x = a, y = c, n = 3.

Derivation of a solution in case of n = 3:

$\boldsymbol{S}(3, \boldsymbol{a}, \boldsymbol{c}) \overset{1}{\Rightarrow} \boldsymbol{A}(3, \boldsymbol{a}, \boldsymbol{c}) \overset{2}{\Rightarrow} \boldsymbol{A}(2, \boldsymbol{a}, \boldsymbol{b})\boldsymbol{p}(3, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{A}(2, \boldsymbol{b}, \boldsymbol{c})$

$\overset{2}{\Rightarrow} \boldsymbol{A}(1, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(2, \boldsymbol{a}, \boldsymbol{b})\boldsymbol{A}(1, \boldsymbol{c}, \boldsymbol{b})\boldsymbol{p}(3, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{A}(2, \boldsymbol{b}, \boldsymbol{c})$

$\overset{3}{\Rightarrow} \boldsymbol{p}(1, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(2. \boldsymbol{a}, \boldsymbol{b})\boldsymbol{A}(1, \boldsymbol{c}, \boldsymbol{b})\boldsymbol{p}(3, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{A}(2, \boldsymbol{b}, \boldsymbol{c})$

$\overset{3}{\Rightarrow} \boldsymbol{p}(1, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(2, \boldsymbol{a}, \boldsymbol{b})\boldsymbol{p}(1, \boldsymbol{c}, \boldsymbol{b})\boldsymbol{p}(3, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{A}(2, \boldsymbol{b}, \boldsymbol{c})$

$\overset{2}{\Rightarrow} \boldsymbol{p}(1, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(2, \boldsymbol{a}, \boldsymbol{b})\boldsymbol{p}(1, \boldsymbol{c}, \boldsymbol{b})\boldsymbol{p}(3, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{A}(1, \boldsymbol{b}, \boldsymbol{a})\boldsymbol{p}(2, \boldsymbol{b}, \boldsymbol{c})\boldsymbol{A}(1, \boldsymbol{a}, \boldsymbol{c})$

$\overset{3}{\Rightarrow} \boldsymbol{p}(1, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(2, \boldsymbol{a}, \boldsymbol{b})\boldsymbol{p}(1, \boldsymbol{c}, \boldsymbol{b})\boldsymbol{p}(3, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(1, \boldsymbol{b}, \boldsymbol{a})\boldsymbol{p}(2, \boldsymbol{b}, \boldsymbol{c})\boldsymbol{A}(1, \boldsymbol{a}, \boldsymbol{c})$

$\overset{3}{\Rightarrow} \boldsymbol{p}(1,\boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(2,\boldsymbol{a}, \boldsymbol{b})\boldsymbol{p}(1, \boldsymbol{c}, \boldsymbol{b})\boldsymbol{p}(3, \boldsymbol{a}, \boldsymbol{c})\boldsymbol{p}(1, \boldsymbol{b}, \boldsymbol{a})\boldsymbol{p}(2, \boldsymbol{b}, \boldsymbol{c})\boldsymbol{p}(1, \boldsymbol{a}, \boldsymbol{c}).$