CSCI5240 – Combinational Search and Optimization with Constraints
Fall 2015
Assignment 2

1155070976 ZHANG Wanying (Sophie)

1.

(a) Give an example of a CSP that is node-consistent but not arc-consistent.

Variable: X, Y

Domain: D(X) = D(Y) = { 1, 2 }

Constraint: X < Y

(b) Give an example of a CSP that is arc-consistent but not node-consistent.

Variable: X, Y

Domain: D(X) = { 0, 1 }, D(Y) = { 2 }

Constraint:

X < Y,

X > 0

(c) Give an example of a CSP that is satisfiable but not node-consistent.

Variable: X, Y

Domain: D(X) = { 0, 1 }, D(Y) = { 2 }

Constraint:

X < Y,

X > 0

(d) Give an example of a CSP that is unsatisfiable but node-consistent.

Variable: X, Y

Domain: D(X) = { 1 }, D(Y) = { 2 }

Constraint:

X > Y,

X > 0

(e) Give an example of a CSP that is satisfiable but not arc-consistent.

Variable: X, Y

Domain: D(X) = { 1, 2 }, D(Y) = { 2 }

Constraint: X < Y

(f) Give an example of a CSP that is unsatisfiable but arc-consistent.

Variable: X, Y

Domain: D(X) = { 1 }, D(Y) = { 2 }

Constraint:

X < Y,

X < 0

(g) Give an example of a CSP that is satisfiable but not node-consistent and not arc-consistent.

Variable: X, Y

Domain: D(X) = { -1, 1, 2 }, D(Y) = { 2 }

Constraint:

X < Y,

X > 0

(h) Give an example of a CSP that is unsatisfiable but node-consistent and arc-consistent.

Variable: X, Y, Z

Domain: D(X) = { 1 }, D(Y) = { 2 }, D(Z) = { 5 }

Constraint:

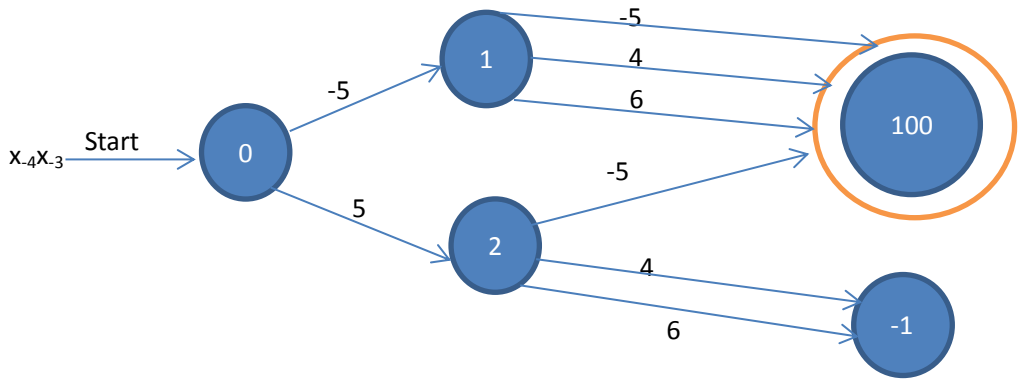
X < Y,

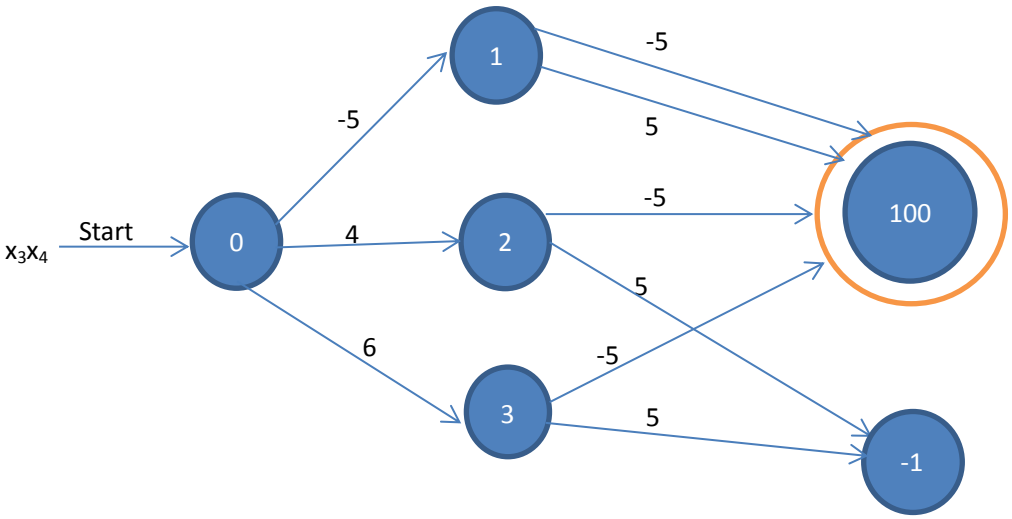
X > 0,

X +Y=Z

2. The n-Queens Problem and the Peaceably Co-existing Armies of Queens

(a)





(b)

Variable: W[1..n][1..n], B[1..n][1..n]

For any integer i ∈[1..n], j ∈[1..n]

W[i][j] means whether put a white queen on the cell of the i_{th} row and j_{th} column of the chessboard,

B[i][j] means whether put a black queen on the cell of the i_{th} row and j_{th} column of the chessboard,

Domain: D(W) = {0, 1}, D(B) = {0, 2}

W[i][j]=1 means put a white queen on the cell of the i_{th} row and j_{th} column of the chessboard,

W[i][j]=0 means don't put a white queen on the cell of the i_{th} row and j_{th} column of the chessboard

B[i][j]=2 means put a black queen on the cell of the i_{th} row and j_{th} column of the chessboard,

B[i][j]=0 means don't put a black queen on the cell of the i_{th} row and j_{th} column of the chessboard

Constraints:

Constraint 1: same-size constraint

(the number of white queens equals to the number of black queens)

$\sum_{i,j} W[i][j] = \left(\sum_{i,j} B[i][j]\right)/2$

(reason of divides 2: if a black queen was put on the cell , the cell would be labeled 2 , thus , the number of black queens equals to $\left(\sum_{i,j} B[i][j]\right)/2$)

Constraint 2: Each row is occupied by at most one type of queen

For every integer i ∈[1..n], j ∈[1..n]

For every integer k ∈[1..n]

W[i][j] * B[i][k] = 0

Constraint 3: Each column is occupied by at most one type of queen

For every integer i ∈[1..n], j ∈[1..n]

For every integer k ∈[1..n]

W[i][j] * B[i][k] = 0

Constraint 4: Each diagonal is occupied by at most one type of queen

For every integer i ∈[1..n], j ∈[1..n]

For every integer x ∈[1..n], y ∈[1..n]

If ((i - j = x – y) or (i + j = x + y))

W[i][j] * B[x][y]==0

Ps: (i - j = x – y) means the coordinates (i,j) and (x,y) share the same positive diagonal,

(i + j = x + y) means the coordinates (i,j) and (x,y) share the same negative diagonal

3. The Knight's Tour

The chessboard labeled as followings: (for n =6)

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

4.

For CSP (Z, D, C):

Z = {x, y, z},

$D_x = D_y = D_z = \{1, 2, 3, 4\}$

$C = \{c_{x,y,z}\}$, where $c_{x,y,z}$ is "x + y = z"

We can introduce a new variable U, (it can be called **encapsulated variable**), U has be assigned a domain that is the Cartesian product of all values can satisfy x + y = z of individual variables x, y, z . Now, the old CSP with 3 variables can be converted to an equivalent unary CSP.

The new equivalent CSP can be written as following,

Variables: U

Domain: $D_U = \{(1,1,2), (1,2,3), (1,3,4), (2,1,3), (2,2,4), (3,1,4)\}$

Constraint: $x=1^{st}(U)$, $y=2^{nd}(U)$, $z=3^{rd}(U)$

For CSP (Z, D, C):

Z = {u, v, x, y},

$D_u = D_v = D_x = D_y = \{1, ..., 100\}$

$C = \{c_{u,v,x,y}\}$, where $C = \{c_{u,v,x,y}\}$ is "u+v = x+y".

We can introduce a new variable W, (it can be called **encapsulated variable**), W has be assigned a domain that is the Cartesian product of all values can satisfy u+v = x+y of individual variables, u, v, x, y. The value that variable W can take is a tuple with four arguments. It can be written like $W = (w_1, w_2, w_3, w_4)$. Now, the old CSP with 4 variables can be converted to an equivalent unary CSP.

The new equivalent CSP can be written as following,

Variables: W

Domain: $D_W = \{(w_1,w_2,w_3,w_4)\}$, where $w_1 \in \{1..100\}$, $w_2 \in \{1..100\}$, $w_3 \in \{1..100\}$, $w_4 \in \{1..100\}$ and $w_1+w_2=w_3+w_4$

Constraint: $u=1^{st}(W)$, $v=2^{nd}(W)$, $x=3^{rd}(W)$, $y=4^{th}(W)$

Further, we can turn a general CSP into an equivalent binary CSP. If the original CSP has only one constraint, we can easily turn it into an equivalent unary CSP using **encapsulated variable**. If the original CSP has more than two constraints, it remains to combine these individual solutions to the solution of the constraint system.

Usually, there are two ways to bind encapsulated variables. One is called hidden variable encoding,

and the other is called dual encoding.

hidden variable encoding (With original variables)

Combination of the original and encapsulated variables naturally expresses the original non-binary CSP as the encapsulated variables directly correspond to the original constraints. Newly introduced constraints just bind the original and encapsulated variables in a following way:

X = i_th_argument_of(U),

where X is the original variable, U is the encapsulated variable and i is the "position of X within U".

For example,

The original CSP:

Variables: X, Y, Z

Domains: D(X) = {1,2}, D(Y) = {3,4}, D(X) = {5,6}

Constraints: X + Y = Z, X < Y

The equivalent CSP:

Variables: U

Domains: D(U) = { (1,4,5), (2,3,5), (2,4,6) }

Constraints: $X=1^{st}(U)$, $Y=2^{nd}(U)$, $Z=3^{rd}(U)$

dual encoding (Without original variables)

The other approach to represent converted binary CSP is based on using encapsulated variables only. Then, the newly introduced constraints just binds these variables via common components in a following way:

i_th_argument_of(U)=j_th_argument_of(V),

where U and V are encapsulated variables and i and j respectively are the "positions of common component".

For example,

The original CSP:

Variables: X, Y, Z

Domains: D(X) = {1,2}, D(Y) = {3,4}, D(X) = {5,6}

Constraints: X + Y = Z, X < Y

Variables: U, V

Domains: D(U) = { (1,4,5), (2,3,5), (2,4,6) }

D(V) = {(1,3), (1,4), (2,3) ,(2,4) }

Constraints: $1^{st}(U)= 1^{st}(V)$, $2^{nd}(U)= 2^{nd}(V)$

My observation: Any CSP can be turned into a binary CSP using **encapsulated variable.** The domain of the encapsulated variable is the solution set of original CSP.

Reference: http://ktiml.mff.cuni.cz/~bartak/constraints/binary.html