XXXday, May XX, 2003
00.00a.m. - 00.00a.m.

University of Glasgow

DEGREES OF M.Eng., B.Eng., B.Sc., M.A. and M.A. (Social Sciences)

COMPUTING SCIENCE 4
CONSTRAINT PROGRAMMING4

(Answer any 2 questions out of 3.)

1. (a) Given a constraint satisfaction problem (V,D,C) describe what we mean by arc-consistency. In your description you can assume that all the constraints in C are binary.

[4]

(b) Describe the AC3 algorithm, using a pseudo-code of your own choosing.

[10]

(c) Compare AC3 to AC5, stating the most significant differences between these two algorithms, and the most significant advantage brought about by AC5.

[4]

(d) Given the choco code below, what are the domains of the variables after the call to propagate(p)? Explain your answer.

```
[csp1() : void
 -> let p := makeProblem("csp1",3),
        x := makeIntVar(p,"x",1,10),
        y := makeIntVar(p,"y",1,10),
        z := makeIntVar(p,"z",1,10)
    in (post(p,x < y - 2),
         post(p,y == 3 * z),
         post(p,z < 2 * x + 1),
         propagate(p),
         for v in p.vars printf("~S ",v))]
```

[3]

(e) In the choco code below, we have made the constraint between variables y and z explicit. What will be the domains of the variables after the call to propagate(p)? Explain your answer.

```
[csp2() : void
 -> let p     := makeProblem("csp2",3),
        x     := makeIntVar(p,"x",1,10),
        y     := makeIntVar(p,"y",1,10),
        z     := makeIntVar(p,"z",1,10),
        goods := list(tuple(3,1),tuple(6,2),tuple(9,3))
    in (post(p,x < y - 2),
        post(p,binRelation(y,z,goods)),
        post(p,z < 2 * x + 1),
        propagate(p),
        for v in p.vars printf("~S ",v))]
```

[4]

2. In the 0-1 Knapsack Problem we are to pack a knapsack with a set of items. There are n distinct items, each with a weight and a value. The knapsack has a capacity X, and the problem is then to select items to pack in the knapsack such that their combined weight is less than or equal to X, and their combined value is as great as possible. The following is an example instance of the problem:

Item:	A	B	C	D	E	F	G	H	I
Value:	13	8	11	16	4	1	1	9	10
Weight:	34	21	19	45	12	2	6	12	22

X = 100

(a) Propose an encoding for the knapsack problem as a constraint satisfaction problem, suggesting the variables you would use, their domains, and the constraints. Note: you do not need to write choco code.

[15]

(b) Suggest possible variable and value ordering heuristics for your encoding and explain why you think your heuristics would be beneficial.

[5]

(c) Explain how constraint programming addresses optimisation. That is, how can we cast an optimisation problem as one of satisfaction.

[5]

3. For a constraint satisfaction problem, the order in which we select variables for instantiation (i.e. the instantiation order) can have a profound effect on search effort. We might classify variable ordering heuristics into two groups, static and dynamic.

(a) Describe the static variable ordering heuristics MBO (minimum bandwidth ordering) and MAXDEG (maximum degree), and suggest why we might expect these to be good heuristics.

[8]

(b) Describe the dynamic variable ordering heuristic SDF (smallest domain first) and suggest why we might expect it to be a good heuristic.

[4]

(c) Are the SDF and MBO heuristics algorithm independent? That is, will SDF and MBO give an advantage to any complete search algorithm we might use to solve a constraint satisfaction problem? Explain your answer.

[4]

(d) Are the SDF and MAXDEG heuristics problem independent? For example, will SDF and MAXDEG give an advantage on all constraint satisfaction problems? Explain your answer.

[4]

(e) On average, the SDF heuristic performs equally well on solvable and unsolvable problems. Why might that be?

[3]

(f) Geelen's value ordering heuristic, *promise*, selects a value that has maximum support in the domains of adjacent variables. Will Geelen's promise reduce the effort for a complete search algorithm when a problem is unsolvable? Explain your answer.

[2]