CS206 Homework #2

Max marks: 60 **Due April 1, 2005**

- *Be brief, complete and stick to what has been asked.*
- *If needed, you may cite results/proofs covered in class without reproducing them.*
- *In this homework, problems* 1 *and* 2 *will be graded, while problems* 3 *and* 4 *will not be graded. Please* ***DO NOT*** *turn in solutions of problems* 3 *and* 4*. They are meant for your own practice.*
- ***Do not copy solutions from others or indulge in unfair means.***

1. *[30 marks]* In a sleepy little town, there is an old postman who *must* visit each morning all houses/offices that have letter(s) to be delivered that day. There are N houses/offices in the town, and each house/office has a distinct address that is a natural number in the range $[1, N]$. The post office itself has the address P.

   The postman, being old, must figure out the "best" sequence of addresses for delivering letters each morning, so that the *total distance* travelled by him *from the post-office to the last address where a letter is delivered and back to the post-office* is minimized. It is assumed that if the postman is currently at address x_i and the next address in the sequence is x_j, then he always takes the shortest path from x_i to x_j. Note that the "best" sequence for delivering letters need not be unique, and there may be letter(s) addressed to the post-office itself.

   Interestingly, this is no ordinary post-office! It has recently acquired a software that can analyze certain classes of predicate logic sentences (formulae without free variables) and can compute models of satisfiable sentences using natural numbers (i.e., the domain of elements is the set of natural numbers). Our old postman, having realized the fun of playing with predicate logic, now wishes to use this software for finding the best sequence of delivery addresses each morning. We must help him in this noble endeavour. In particular, we must formulate a satisfiable predicate-logic sentence ϕ (containing no free variables), such that the "best" sequence of addresses can be "extracted" from a model of ϕ using natural numbers. In constructing ϕ, we are allowed to use the following predefined predicates and functions:

   - $\mathsf{L}(x_i)$: This predicate takes a natural number x_i as argument, and if x_i lies in the range $[1, N]$, then it evaluates to True iff there is a letter for address x_i. If x_i lies outside the range $[1, N]$, the predicate evaluates to False. Clearly, the definition of $\mathsf{L}(x_i)$ potentially changes every morning (depending on which addresses have letters to be delivered that day). Our postman, in his enthusiasm to use the software, has agreed to look at the pile of letters each morning and update the definition of this predicate on the post-office computer before running the software on ϕ.
   - $\mathsf{D}(x_i, x_j)$: This function takes two natural numbers and returns a natural number giving the shortest distance between addresses x_i and x_j in the town, for all $x_i, x_j \in \{1 \ldots N\}$. If either argument is outside the range $[1, N]$, the function returns (rather arbitrarily chosen) 13. The town is seismologically stable, and there are no new constructions or demolitions. So the definition of this function is assumed to be time invariant, and does not change from one day to another

- $+(x_i, x_j)$: This function takes two natural numbers and returns their sum.
- $*(x_i, x_j)$: This function takes two natural numbers and returns their product.
- $x_i = x_j$: This is the equality predicate for natural numbers.
- $x_i \geq x_j$: This is the greater-than-or-equal-to predicate for natural numbers.

In addition to the above predefined predicates and functions, you are allowed to use the function symbol Seq of arity 1 (from natural numbers to natural numbers) in formulating ϕ. No other function or predicate symbols may be used in ϕ.

Recall that our noble goal was to to help the old postman "extract" the "best" sequence of addresses every morning. So, we must use the function symbol Seq in ϕ in such a way that if the post-office's software returns a model $M = (\mathbf{N}, \mathsf{Seq}^M)$ for ϕ (where $\mathbf{N}$ is the set of natural numbers and $\mathsf{Seq}^M : \mathbf{N} \rightarrow \mathbf{N}$ is a concrete function definition), then Seq^M can be used to obtain a "best" sequence of addresses for delivering letters. In particular, if there are K addresses that have letters to be delivered on a particular day, then $\mathsf{Seq}^M(j)$ must give the j^{th} address in a "best" sequence for that day, for all $j \in [1, K]$. For $j \notin [1, K]$, $\mathsf{Seq}^M(j)$ must be 0. You may assume that there is at least one letter to be delivered each day, i.e. $1 \leq K \leq N$ for each day, although K may vary from day to day.

Note that ϕ must not involve K, since this may indeed differ from day to day, and our old postman cannot enter a new formula to the software each day. N is assumed to be fixed, so you may use N and functions of N in formulating ϕ, but once formulated, our postman must be able to use the same formula ϕ every day, simply by changing the definition of $\mathsf{L}(x_i)$ each morning as described above.

As an example, suppose there are 20 houses/offices in the town, and on a particular day, there are letters for addresses $1, 2$ and 12. Suppose also that the unique best sequence of addresses (for minimizing the total distance travelled) is $(12, 2, 1)$. To find this sequence, the postman goes and updates the definiton of L so that $\mathsf{L}(1) = \mathsf{L}(2) = \mathsf{L}(12) = \mathsf{True}$ and $\mathsf{L}(x_i) = \mathsf{False}$ for all other x_i. Our formula ϕ should be such that with the above definiton of L, a model $M = (\mathbf{N}, \mathsf{Seq}^M)$ of ϕ has $\mathsf{Seq}^M(1) = 12, \mathsf{Seq}^M(2) = 2, \mathsf{Seq}^M(3) = 1$ and $\mathsf{Seq}^M(i) = 0$ for all other i.

Give a predicate logic sentence ϕ for the above purpose, and explain briefly the justification behind the construction of your formula. In your justification, you must argue why any model $(\mathbf{N}, \mathsf{Seq}^M)$ of ϕ must contain a function Seq^M that gives a "best" sequence of addresses.

Important: Your justification **must not** exceed 500 words, and your formula **must** have size linear in N. In other words, the total no. of symbols (variables, predicate-logic operators, function and predicate symbols, parentheses) in your formula must be $O(N)$.

2. *[10+10+10 marks]* Prove the following sequents using natural deduction for predicate logic:

   (a) $\exists x\, \exists y\, (H(x, y) \vee H(y, x)),\ \neg\exists x\, H(x, x) \vdash \exists x\, \exists y\, \neg(x = y)$.

   (b) $\forall x\, P(a, x, x),\ \forall x\, \forall y\, \forall z\, (P(x, y, z) \rightarrow P(f(x), y, f(z))) \vdash \exists z\, P(f(a), z, f(f(a)))$.

   (c) $\forall x\, \exists y\, P(x, y), \forall x\, \forall y\, P(x, y) \rightarrow \neg P(y, x) \vdash \neg\exists z\, \forall u\, P(u, z)$.

   Important: None of your proofs should exceed twenty-five steps of application of basic introduction and elimination rules.

3. ***[Ungraded problem. Do not turn in solutions to this problem.]*** Let $\mathcal{X} = \{x_1, x_2, \ldots\}$ be a countably infinite set of variables, $\mathcal{P} = \{P_1, P_2, \ldots\}$ be a countably infinite set of predicate symbols, and $\mathcal{F} = \{f_1, f_2, \ldots\}$ be a countably infinite set of function symbols. We consider the set of all predicate logic sentences that can be constructed using the above symbols, parentheses, commas, and standard predicate logic operators, i.e., $\{\neg, \vee, \wedge, \exists, \forall\}$. Thus, our sentences of interest are simply finite strings of symbols from the set $\mathcal{X} \cup \mathcal{P} \cup \mathcal{F} \cup \{(,), \neg, \vee, \wedge, \exists, \forall, ,\}$, with appropriate syntactical structure. Note that the arities of predicate and function symbols have not been pre-defined. Instead, these are defined by the way in which these symbols are used in constructing a predicate logic sentence. Thus, the same symbol can be used in two different sentences with two different arities.

   As an example, $\phi_1 = \exists x_1\, P_1(f_1(x_1))$ and $\phi_2 = \forall x_1\, \forall x_2\, P_1(x_1, x_2) \;\rightarrow\; \exists x_1\, \forall x_2 P_1(x_1, x_2)$ are two sentences constructed from the above symbols. Here, we have used P_1 with arity 1 in ϕ_1 but with arity 2 in ϕ_2; nevertheless both of them are predicate logic sentences using symbols from the above sets. We will assume that in constructing a sentence, if we need k distinct variables, l distinct predicate symbols and m distinct function symbols, then we will use the variables $\{x_1, \ldots x_k\}$, predicate symbols $\{P_1, \ldots P_l\}$ and function symbols $\{f_1, \ldots f_m\}$. Note that this doesn't reduce the generality of of predicate logic sentences that can be constructed.

   Our goal in this question is to reason about the existence or non-existence of a computer program that

   - Prints *every valid* predicate logic sentence that can be constructed using the above rules, and
   - Prints *nothing else*.

   Clearly the set of valid predicate logic sentences that can be constructed using the above rules is countably infinite. Thus, our program must be a non-terminating program. Nevertheless, we wish to guarantee that every valid predicate logic sentence constructed using the above rules is generated by our program after a finite number of steps. In particular, we wish to give an upper bound on the number of steps within which a given valid predicate logic formula will be generated by our program.

   (a) Show that such a program indeed exists. You must give a constructive argument, indicating the sequence of steps that the program iterates through in order to achieve its intended purpose. You may assume that integers of unbounded size can be represented on the computer in which this program runs.

   (b) Given a valid predicate logic sentence ϕ, describe how you would obtain an upper bound on the number of steps required by your program to print this sentence.

4. ***[Umgraded problem. Do not turn in solutions to this problem.]*** Use natural deduction to prove the following sequents:

   (a) $\forall x\, (P(x) \rightarrow (x = b)), \forall x\, (x = b) \rightarrow P(x) \;\vdash\; P(b) \wedge \forall x\, \forall y\, (P(x) \wedge P(y) \rightarrow (x = y))$.

   (b) $\forall x\, (P(x) \vee Q(f(x))), \forall x\, (R(f(x)) \rightarrow \neg P(x)), \exists x\, R(f(x)) \;\vdash\; \exists x\, Q(f(x))$.

   (c) $\forall x\, \forall y\, P(x, y) \wedge P(y, x) \rightarrow (x = y), \forall x\, P(x, f(x)), \exists z\, \forall x\, P(x, z) \;\vdash\; \exists u\, (u = f(u))$.