

Part II

Exercise 1 (No notconnected) [3 pts] Consider representing the finite graph $G = (N, E)$, with nodes N and edges $E \subseteq N \times N$, as the set of atomic formulas

$$\mathcal{G} = \{\text{node}(x) \mid x \in N\} \cup \{\text{edge}(x, y) \mid \langle x, y \rangle \in E\}.$$

Argue why it is impossible to write a logic program \mathcal{P} in first-order hereditary Harrop formulas that specifies the predicate $\text{nc}(x, y)$ such that for all $x, y \in N$, x and y are not connected by a path in the graph G if and only if the sequent $\mathcal{G}, \mathcal{P} \vdash \text{nc}(x, y)$ is provable.

Exercise 2 (A linear and exponential sized proof) [3 pts] Let a_1, a_2, a_3, \dots be a denumerable sequence of propositional symbols. Let D_i be the propositional Horn clause $D_i = (\bigwedge_{j=1}^{i-1} a_j) \supset a_i$. Thus, $D_1 = a_1$, $D_2 = a_1 \supset a_2$, $D_3 = (a_1 \wedge a_2) \supset a_3$, etc. It is easy to see that the sequent

$$D_1, D_2, \dots, D_n \vdash a_n$$

is provable. Using the proof system for classical and intuitionistic logic (p. 19), find two cut-free proofs for this sequent. One should have size at-most quadratic in n and one should have size exponential in n . Here, size refers to the total number of occurrences of inference rules in a proof.

Exercise 3 (Positive and negative connectives) [3 pts] Some of the linear logic connectives are divided into positive connectives, namely, $\mathbf{1}$, $\mathbf{0}$, \otimes , \oplus , \exists and the negative connectives, namely, \perp , \top , \wp , $\&$, \forall . Let B and C be two formulas for which $B \equiv !B$ and $C \equiv !C$. Show that the following equivalences hold for the positive connectives.

$$\mathbf{1} \equiv !\mathbf{1} \quad \mathbf{0} \equiv !\mathbf{0} \quad B \otimes C \equiv !(B \otimes C) \quad B \oplus C \equiv !(B \oplus C) \quad \exists x.B \equiv !\exists x.B$$

If instead B and C are two formulas such that $B \equiv ?B$ and $C \equiv ?C$. Show that the following equivalences hold for the negative connectives.

$$\perp \equiv ?\perp \quad \top \equiv ?\top \quad B \wp C \equiv ?(B \wp C) \quad B \& C \equiv ?(B \& C) \quad \forall x.B \equiv ?\forall x.B$$

Here, the equivalence $B \equiv C$ means that the formula $(B \multimap C) \& (C \multimap B)$ is provable in linear logic.

Exercise 4 (Hiding an intermediate) [3 pts] Consider the following two linear logic formulas:

$$\exists a_2. \left\{ \begin{array}{l} (a_1 \wp m_1 \multimap a_2 \wp m_2) \otimes \\ (a_2 \wp m_3 \multimap a_3 \wp m_4) \end{array} \right\} \dashv\vdash (a_1 \wp m_1) \multimap (m_2 \multimap (m_3 \multimap (a_3 \wp a_4)))$$

Operationally, the formula on the left can be viewed as describing how an agent, say Alice, may interact with her environment: if Alice at step 1 inputs message m_1 then she can output message m_2 and moves to step 2; and if Alice at step 2 inputs message m_3 then she can output message m_4 and move to step 3. The existential quantifier can be used to formally hide the second step of Alice (no one can synchronize directly with her in step 2). Formally prove that the formula on the left and the formula on the right are logically equivalent in linear logic. Treat higher-order quantification explicitly: that is, carefully state what substitutions are used, if any, for the propositional variable a_2 .

Exercise 5 (3 pts) Consider representing the finite graph $G = (N, E)$, with nodes N and edges $E \subseteq N \times N$, as the set of formulas

$$\mathcal{G} = \{\text{node}(x) \mid x \in N\} \cup \{!\text{edge}(x, y) \mid \langle x, y \rangle \in E\}$$

(notice the use of!). Consider the logic program \mathcal{P} that consists of the following three formulas.

$$\forall u[\text{connected} \multimap (\text{node}(u) \otimes (\text{nd}(u) \Rightarrow \text{loop}))].$$

$$\text{loop} .$$

$$\forall u, v[\text{loop} \multimap (\text{nd}(u) \otimes \text{edge}(u, v) \otimes \text{node}(v) \otimes (\text{nd}(v) \Rightarrow \text{loop}))].$$

Show that the sequent $\mathcal{G}, \mathcal{P} \vdash \text{connected}$ is provable in linear logic (Lolli) if and only if the graph G is connected.