Course: Structural Proof Theory and Abstract Logic Programming (SPAL)
Lecturer: Dr Paola Bruscoli
ICCL/Institute for Artificial Intelligence/Computational Logic
Summer Semester 2011

Coursework for SPAL – due 20/07/2011

General Notes:
This coursework is meant to be individual work to be carried out of lecturing and tutorial times.
Its assessment will contribute to the final grade for the course.
Moreover, purpose of this coursework, is to offer students the possibility of a self-assessment, on questions of comparable nature that could be asked during an examination.
Please print this page, sign the declaration below, filling all the remaining fields, including the date of submission.
You may add as many sheets for your solutions as you want, be sure that all your solutions to the single exercises are numbered accordingly.
Before submitting your coursework, please make sure that all sheets are adequately and firmly stapled together.
Deadline for submission is strict; you may submit your work by posting it in the white mailbox out of room 2001.
Good luck!

Declaration:
Hereby, I, (name/family name) ............................................................................
having student registration number ..............................................................................,
declare that the solution to this coursework is the result of individual work, and that I am the only author of it.
I have read the general notes and I have submitted a total of ............ sheets, including the present one; all sheets are stapled.

Date of submission ............................... Signature..........................................

1
Exercise 1 [points: 4]
Consider Hilbert-Tarski system HT, based on the language of implication and negation. Can you use it to prove conjunctive and disjunctive formulae? Explain your reasons.

Exercise 2 [points: 4]
Consider the following axiomatic system X:
1. \( A \supset (B \supset A) \)
2. \( (A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C)) \)
3. \( (A \supset (B \supset C)) \supset (B \supset (A \supset C)) \)
4. \( (A \supset B) \supset (\neg B \supset \neg A) \)
5. \( \neg \neg A \supset A \)
6. \( A \supset \neg \neg A \)
How would you proceed to prove that X and HT prove the same theorems? Give a sketch of your reasoning.

Exercise 3 [points: 4]
Consider the system for natural deduction for intuitionistic logic, have called BHK. State the principles that have been followed for designing the introduction and the elimination rules.

Exercise 4 [points: 1+4+3+2]
With reference to the systems \( N_I \) or \( N_C \) (sequent style presentation of natural deduction), answer the following questions:

1. Define the notion of sequents in these calculi;
2. Explain the advantages of using a sequent style presentation for natural deduction;
3. Prove in \( N_I \) or \( N_C \) that \( (A \supset (A \supset B)) \supset (A \supset B) \) is a theorem.
4. Discuss if sequents in \( N_I \) or \( N_C \) obey the subformula property.

Exercise 5 [points: 4]
Consider sequent calculus proof systems for intuitionistic and classical logic, \( G_I \) and \( G_C \); explain why structural rules are needed.

Exercise 6 [points: 4]
Consider sequent calculus proof systems for classical logic \( G_C \) and \( LK \); give a proof in both systems for the formula \( (\neg B \supset \neg A) \supset ((\neg B \supset A) \supset B) \).

Exercise 7 [points: 6]
Define a transformation from proofs in \( G_C \) to proofs in \( LK \).

Exercise 8 [points: 1+3]
Can we transform any deduction in \( N_C \) in a cut-free proof of \( G_C \)? Explain why.

Exercise 9 [points: 4]
Discuss two uses of a cut-elimination theorem for a proof system.

Exercise 10 [points: 4]
In the cut-elimination proof for \( LK \), show how the cut is eliminated when you have the following case: the cut formula is an implication \( A \supset B \) which is active in some logical rule.
**Exercise 11** [points: 4]
Suggest the reasons to introduce a multicut in the cut-elimination technique by Tait.

**Exercise 12** [points: 2]
State the shape of a cut-rule in additive form and in multiplicative form.

**Exercise 13** [points: 2]
Suggest how to transform an additive rule for conjunction on the left into an equivalent multiplicative one, by using structural rules.

**Exercise 14** [points: 2]
Suggest why the rule of contraction is admissibile in systems of the $G_3$ family (for example, in intuitionistic logic).

**Exercise 15** [points: 2+2]
Identify the pairs of dual connectives in the following lists, referring to linear logic: tensor, par, plus, with. State which ones are additive and which one are multiplicative.

**Exercise 16** [points: 4]
Explain the concept of uniform provability.

**Exercise 17** [points: 4]
In system MNPS, give an example of a formula that can be proven but cannot be uniformly proven and show your proofs.

**Exercise 18** [points: 3]
Discuss how you can recover a form of completeness aiming in maintaining uniform provability.

**Exercise 19** [points: 6]
Formally define the notion of Abstract Logic Programming Language.

**Exercise 20** [points: 6]
Identify an Abstract Logic Programming Language of your choice and formally prove that it is indeed an Abstract Logic Programming Language.

**Exercise 21** [points: 5]
Give a sequent calculus proof of the following formula of linear logic: $a \perp \otimes (a \otimes a), a \not\otimes (a\perp \otimes a\perp)$.

**Exercise 22** [points: 3]
Draw the proof net associated to the proof you have found in 21).

**Exercise 23** [points: 4]
Explain why the proof net you have found in 22 is correct, i.e. it satisfies Danos-Regnier correctness criterion.

**Exercise 24** [points: 3]
Give an example of a proof net which is not correct, i.e. it doesn’t correspond to a legitimate proof in the sequent calculus.

**Exercise 25** [points: 3]
Give a deep inference proof in the calculus of structures, in system MLL, for the linear logic formula stated in 21).