Modal Logic
Exam questions for MGS 2005

If you want to do an exam question for the Modal Logic course, please do one of the following questions. (If you want to do all three, I am happy to check them.)

Question 1: Proofs in K Prove the following theorems in K:

1. $\Box\phi \wedge \Box\psi \rightarrow \Box(\phi \wedge \psi)$
2. $\Box(\phi \rightarrow \psi) \rightarrow (\Diamond\phi \rightarrow \Diamond\psi)$
3. $\Diamond(\phi \wedge \psi) \rightarrow \Diamond\phi$
4. $\neg\Diamond\bot$

Please note that the deduction theorem:

$$\Gamma, \phi \vdash \psi \quad \Rightarrow \quad \Gamma \vdash \phi \rightarrow \psi$$

does not hold for K. However, it holds so long as the necessitation rule ('from ϕ derive $\Box\phi$') is not applied to any of the assumptions or to formulas dependent on undischarged assumptions.

Question 2: Correspondence and completeness Building on the completeness proof for K given in the lecture, prove that K + $\Diamond\top$ is complete for the class of models where R is serial (every world has a successor: $\forall x \exists y R(x, y)$).

Question 3: Bisimulation Work out the complexity of the algorithm as presented in the lecture and suggest any improvements to it. Write a version of Tarjan's algorithm which does not ignore propositional variables.

Slides of the course I gave in 2003 are on
`http://www.cs.nott.ac.uk/~nza/modal.html`.

Recommended literature:

1. P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2001. **For most systems and logical background.**
2. Sally Popcorn. *First Steps in Modal Logic*. Cambridge University Press, 1994. **For very precise introduction to basic multimodal logic.**
3. M. Huth and M. Ryan. *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge University Press, 2000. **For exposition aimed at computer scientists, easy introduction to model checking.**
4. R. Paige and R. Tarjan. Three partition refinement algorithms. In *SIAM Journal of Comput.* **16**(6), 1987, 973–989. **Bisimilarity checking algorithm.**