Revision Lecture

Computer-Aided Verification

Dave Parker

University of Birmingham

2016/17

Today

- Exam
  - details; revision; syllabus

- Module content summary
  - main topics covered
  - key points/ideas
  - feedback & tips from assignments
  - examples (e.g. from 2016 past paper)

- Questions, interruptions welcome…

Exam

- Final module mark:
  - 20% continuous assessment (see Canvas) + 80% exam
- Exam
  - 1.5 hours (9.30am Thur 11 May)
  - 4 questions (answer all; equal weighting)
  - no appendix/supplementary material
- Revision resources
  - Assignments 1–3 (look carefully at model solutions)
  - also non-assessed exercises with solutions (CTL, BMC)
  - 2016 exam paper (my.bham, Canvas)
  - see also the Baier/Katoen book

Module contents (examinable)

- Modelling sequential and parallel systems
  - labelled transitions systems, parallel composition
- Temporal logic
  - LTL, CTL and CTL*, etc.
- Model checking
  - CTL model checking algorithms
  - automata-theoretic model checking (LTL)
- ~~Verification tools: SPIN~~
- Advanced verification techniques
  - bounded model checking via propositional satisfiability
  - ~~symbolic model checking~~
  - ~~probabilistic model checking~~

Modelling systems

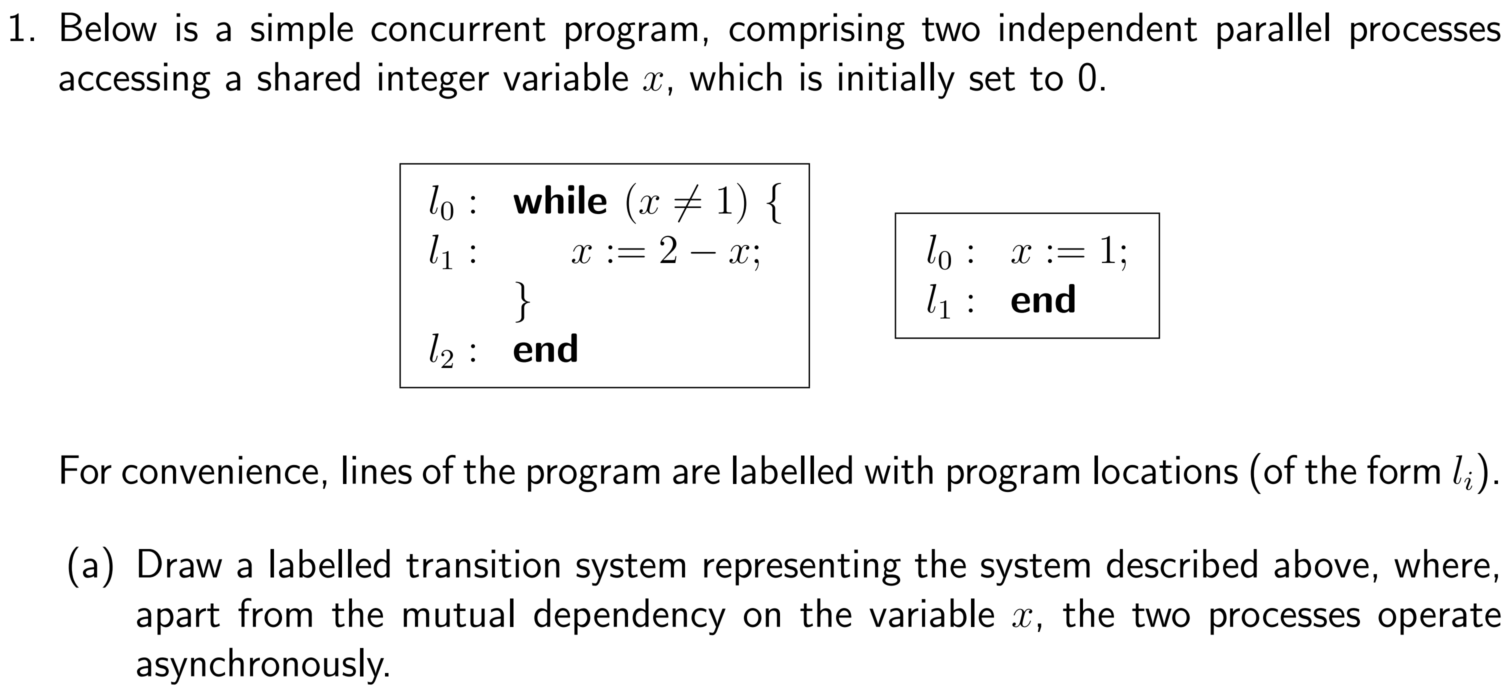
- Modelling sequential systems as LTSs
  - e.g. imperative programs, reactive systems
  - nondeterminism

- Modelling parallel systems: concurrency through interleaving
  - either asynchronous (M_1 ||| M_2) or synchronous (M_1 $||_H$ M_2)
  - also parallel programs/processes with shared variables

- Feedback & tips (Assignment 1)
  - always make clear what the states are
  - states: everything needed to capture dynamic (nothing more)
  - for a product, states should have both component parts of state
  - never duplicate states
  - lay states out logically when drawing LTSs

2016 past paper

- Qu 1a (drawing an LTS)



1. Below is a simple concurrent program, comprising two independent parallel processes accessing a shared integer variable x, which is initially set to 0.

```
$l_0$ :  while ($x \neq 1$) {
$l_1$ :      $x := 2 - x$;
         }
$l_2$ :  end
```

```
$l_0$ :  $x := 1$;
$l_1$ :  end
```

For convenience, lines of the program are labelled with program locations (of the form l_i).

(a) Draw a labelled transition system representing the system described above, where, apart from the mutual dependency on the variable x, the two processes operate asynchronously.

Properties (of LTSs)

- Linear-time properties
  - infinite paths & traces (sequences of states & labellings)
  - properties are just sets of (“good”) traces

- Property classes, informally:
  - invariants: "something good is always true”
  - safety properties: "nothing bad happens" (in finite time)
  - liveness properties: "something good happens in the long run"

- Feedback & tips (Assignment 2)
  - property class (e.g. safety) does not relate to a specific model
  - to justify what class a property is, look carefully at definitions: e.g. invariants (proposition), safety properties (bad prefix), liveness (all finite words extendable)

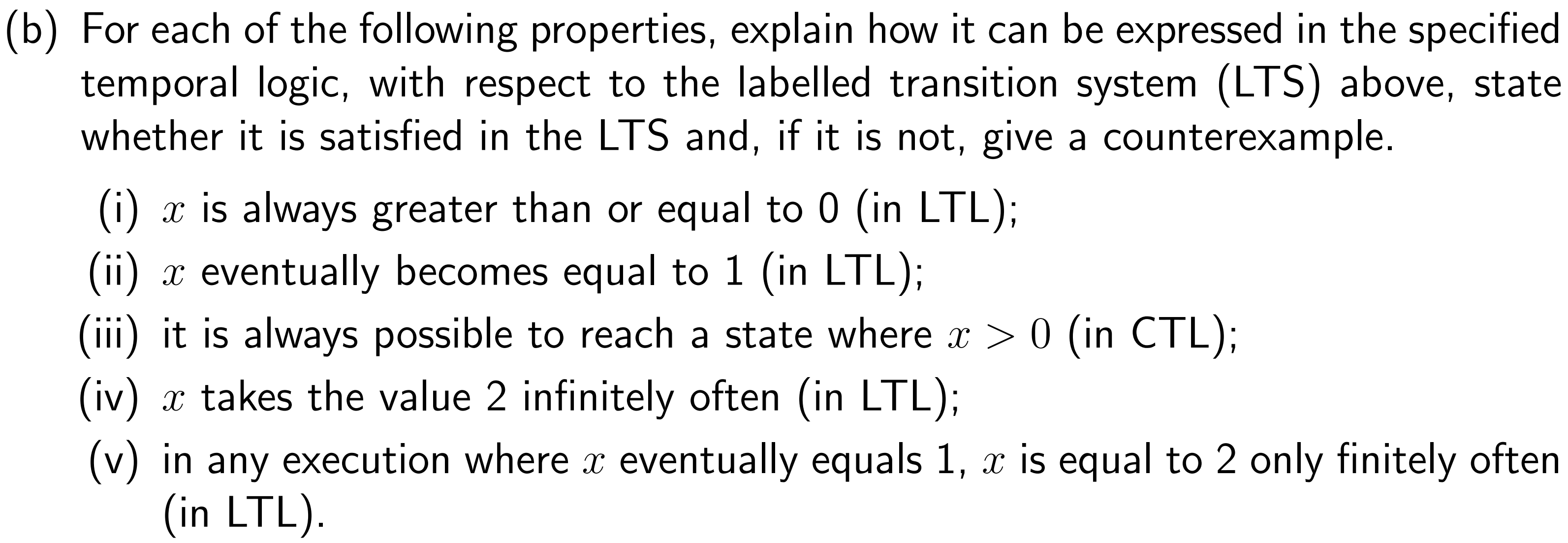
Temporal logic

- Temporal logics: CTL, LTL, etc.
  - propositional logic + temporal operators ($\bigcirc$, U, $\Diamond$, $\Box$)
  - CTL adds existential/universal quantification over paths
  - syntax, semantics, translation, satisfaction in an LTS
  - equivalences: proving, disproving

- Feedback & tips (English to logic)
  - always make clear what the atomic propositions are
  - remember common templates
    - e.g. $\Diamond a$, $\Box \neg b$, $\neg a \mathrm{U} b$, $\Box(a \rightarrow \Diamond b)$, $\Box(a \rightarrow \bigcirc b)$, $\Box \Diamond a$, $\Diamond \Box b$, $\Diamond \Box \neg a$
- Feedback & tips (equivalences)
  - equivalences: again not specific to a model
  - just remember common equivalences/dualities
    - esp. for propositional logic and individual temporal operators

2016 past paper

- Qu 1b (English to temporal logic)



(b) For each of the following properties, explain how it can be expressed in the specified temporal logic, with respect to the labelled transition system (LTS) above, state whether it is satisfied in the LTS and, if it is not, give a counterexample.

(i) x is always greater than or equal to 0 (in LTL);

(ii) x eventually becomes equal to 1 (in LTL);

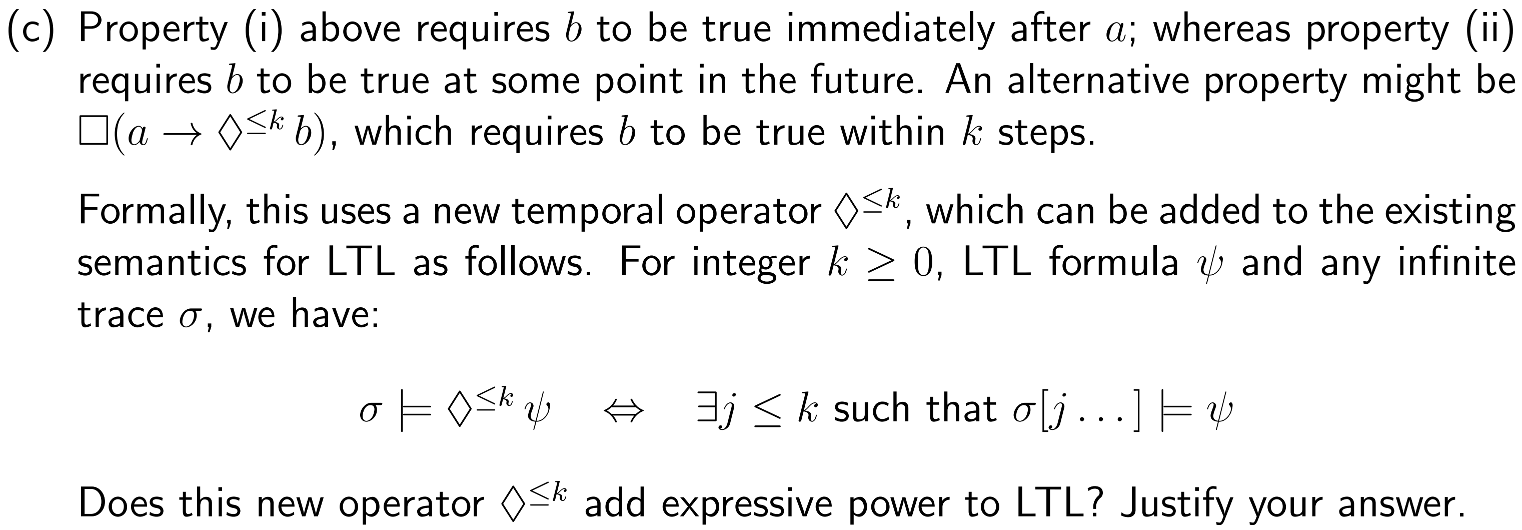
(iii) it is always possible to reach a state where $x > 0$ (in CTL);

(iv) x takes the value 2 infinitely often (in LTL);

(v) in any execution where x eventually equals 1, x is equal to 2 only finitely often (in LTL).

2016 past paper

- Qu 3c (expressive power)



(c) Property (i) above requires b to be true immediately after a; whereas property (ii) requires b to be true at some point in the future. An alternative property might be $\Box(a \rightarrow \Diamond^{\leq k} b)$, which requires b to be true within k steps.

Formally, this uses a new temporal operator $\Diamond^{\leq k}$, which can be added to the existing semantics for LTL as follows. For integer $k \geq 0$, LTL formula ψ and any infinite trace σ, we have:

$$\sigma \models \Diamond^{\leq k} \psi \quad \Leftrightarrow \quad \exists j \leq k \text{ such that } \sigma[j \ldots] \models \psi$$

Does this new operator $\Diamond^{\leq k}$ add expressive power to LTL? Justify your answer.

Model checking

- CTL model checking
  - conversion to existential normal form (ENF)
  - recursive computation of satisfying states Sat(ϕ)
  - basic set operations for propositional logic
  - look at transitions for ∃◯, graph algorithms for ∃U, ∃□

- Feedback & tips (model checking in general)
  - don't forget to validate your answers
  - show your working for all parts of the algorithms
  - if asked, relate back to original model
- CTL:
  - see unassessed exercises (Canvas) for CTL model checking

Automata-based model checking

- Finite automata (NFAs) & regular languages
- Regular safety properties
  - bad prefixes represented by an NFA
  - model checking: reachable accept states in LTS-NFA product
  - $M \nvDash P_{safe} \Leftrightarrow$ some path satisfies $\Diamond$accept in $M \otimes \mathcal{A}$
- Büchi automata (NBAs) & ω-regular languages (e.g. LTL)
  - model checking: accepting cycles in LTS-NFA product
  - $M \nvDash \psi \Leftrightarrow$ some path satisfies $\Box\Diamond$accept in $M \otimes \mathcal{A}_{\neg\psi}$
- Counterexamples/witnesses

Automata-based model checking

- Feedback & tips
  - again: don't forget to validate your answers
  - and, if asked, relate back to original LTS (as well as product)
  - remember automata for common LTL formulae
  - also patterns/recipes in common automata
  - products: layout the LTS logically
  - initial state is not always (s_0,q_0)

Questions

- Questions
  - quickest/easiest on Facebook/email
  - office hour today (3-4pm)
  - further office hours next week (TBA)