EECS3342 System Specification and Refinement
Fall 2026

Digest of Lecture 19, November 19, 2026

New-Event Simulation and Livelock

Jackie Wang

How to use this Digest

Use this digest as a **roadmap and checklist**. It highlights the main ideas, important examples, common pitfalls, practice targets, and where to return in the course materials. It does **not** replace attending or reviewing the session, using the recording when needed, studying Jackie's annotated iPad notes and other official course materials, completing assigned work, or practising independently. Use it to identify what you understand, what you need to revisit, and what you should be able to do on your own.

If time is limited, start with the sections most relevant to your current review and return to the rest later; selective use is better than skipping the digest because it looks long.

This is an AI-assisted study roadmap reviewed by Jackie and grounded in the session transcript and Jackie's annotated iPad notes. Like any course document, it may still contain an error or omission. If something seems technically doubtful or inconsistent with the original course materials, return to those materials and ask Jackie for clarification rather than treating the digest as final authority.

Table of Contents

Main Ideas 2

Roadmap 2

Key Terms, Notation, and Structures 5

Key Examples and Demonstrations 6

Common Pitfalls and Misconceptions 7

What You Should Be Able to Do 8

Suggested Practice 8

Where to Review 9

Closing Reminder 9

Main Ideas

Lecture 19 completed the first-refinement treatment of the two **new** events and used them to motivate a new refinement concern: livelock or divergence. The main movement was:

- determine the guards of `IL_in` and `IL_out`, including the important distinction between a guard that is **correct but redundant** and one that must remain explicit;
- write complete before-after predicates for the new-event actions and use the linking invariant
$$a + b + c = n$$
to observe that either new event leaves the abstract quantity n unchanged;
- represent that abstract-level non-change using an artificial **skip** transition with effect $n' = n$, so a concrete new-event transition can still be simulated at the abstract level;
- formulate invariant-preservation obligations for the new events and see how the effect of `skip` enters a post-state goal such as the linking invariant;
- compare concrete and abstract traces: a finite number of new events corresponds to a finite number of `skip`s, but infinitely many new events can produce infinitely many `skip`s, giving **livelock/divergence** rather than deadlock.

Jackie also encouraged students to begin exam preparation. The detailed method for preventing divergence was deliberately deferred to the next lecture, where a new concept called a **variant** would be introduced.

Roadmap

1. Decide which guards the two new events really need

For `IL_in`, a car moves from the bridge onto the island. Its action is

$$a := a - 1, \qquad b := b + 1,$$

so the event must require

$$a > 0.$$

A second candidate condition, $c = 0$, is logically sensible because traffic in the opposite bridge direction should be absent. However, Jackie showed that it is **redundant**: assuming the model axioms, the pre-state invariants, and $a > 0$, the condition $c = 0$ is provable. In particular, the safety invariant

$$a = 0 \vee c = 0$$

combined with $a > 0$ forces $c = 0$.

For `IL_out`, a car leaves the island and enters the bridge in the opposite direction. Its action is

$$b := b - 1, \qquad c := c + 1.$$

The guard $b > 0$ is necessary so that a car is actually available on the island. Jackie then considered the candidate $a = 0$. Unlike the previous case, $a = 0$ cannot be proved merely from the axioms, invariants, and $b > 0$, so it must remain as an explicit guard:

$$\texttt{IL_out}: \qquad b > 0 \ \wedge \ a = 0.$$

The broader lesson is not to keep or delete a plausible guard by intuition alone. Test whether the candidate guard follows from the other guards together with the axioms and pre-state invariants.

2. Complete the before-after predicates of the new events

Jackie then rewrote the actions as complete before-after predicates, including variables that are unchanged:

$$\begin{aligned} \texttt{IL_in}: &\qquad a' = a - 1 \ \wedge \ b' = b + 1 \ \wedge \ c' = c, \\ \texttt{IL_out}: &\qquad a' = a \ \wedge \ b' = b - 1 \ \wedge \ c' = c + 1. \end{aligned}$$

The unchanged-variable clauses matter when the predicates are substituted into post-state invariants.

3. Use the linking invariant to discover the abstract effect

Before either new event, M1 satisfies

$$a + b + c = n.$$

For `IL_in`, substituting the concrete action into the post-state sum gives

$$(a - 1) + (b + 1) + c = a + b + c = n.$$

For `IL_out`, similarly,

$$a + (b - 1) + (c + 1) = a + b + c = n.$$

Therefore the abstract quantity has the same value before and after either new event:

$$n' = n.$$

This observation identifies the abstract behavior that must correspond to a concrete new-event transition.

4. Introduce the artificial abstract event `skip`

A new event exists only in the concrete refinement, so M0 has no ordinary abstract event with the same name. To visualize simulation, Jackie introduced an **artificial abstract dummy transition** called `skip`. Its effect is simply

$$n' = n.$$

It has no guard in the illustration, so its guard is effectively true. `skip` is a reasoning device for the refinement correspondence; it is **not** an event that was actually added to the M0 model.

The commuting-diagram idea therefore remains the same as for old events: a concrete transition from w to w' must be simulated by an abstract transition from v to v', and the abstract/concrete states must satisfy the refinement invariants before and after. For a new event, the abstract transition is the dummy `skip`.

5. Generate invariant-preservation obligations for new events

M1 has two new events, `IL_in` and `IL_out`, and five concrete invariants. Looking only at the new events therefore gives

$$2 \times 5 = 10$$

invariant-preservation obligations. If the two old concrete events are included as well, M1 has four concrete events in total, giving $4 \times 5 = 20$ preservation obligations across the whole model.

Jackie instantiated `IL_in/inv1_4/INV`. The hypotheses include the axioms, the abstract invariants, the five concrete invariants, and the guard $a > 0$. Starting from the post-state linking invariant

$$a' + b' + c' = n',$$

the concrete event gives

$$a' = a - 1, \qquad b' = b + 1, \qquad c' = c,$$

while the abstract dummy transition gives

$$n' = n.$$

The resulting goal is

$$(a - 1) + (b + 1) + c = n.$$

Jackie left the actual inference-rule discharge as active proof practice.

6. Compare one concrete trip with its abstract trace

Jackie traced one car through the full concrete trip. After initialization, the concrete event sequence is

$$\langle \texttt{init}, \texttt{ML_out}, \texttt{IL_in}, \texttt{IL_out}, \texttt{ML_in} \rangle.$$

The corresponding abstract trace is

$$\langle \texttt{init}, \texttt{ML_out}, \texttt{skip}, \texttt{skip}, \texttt{ML_in} \rangle.$$

The two finite `skip`s are harmless: each simply corresponds to one of the two concrete new-event transitions.

7. Preview livelock/divergence through infinitely many new events

The final example deliberately modified M1 for demonstration. The guards of the new events were removed, making those events always enabled, and other constraints were weakened. This permits an unbounded alternation of concrete new events such as

$$\texttt{IL_in},\ \texttt{IL_out},\ \texttt{IL_in},\ \texttt{IL_out},\ldots$$

that corresponds abstractly to

$$\texttt{skip},\ \texttt{skip},\ \texttt{skip},\ldots$$

forever.

Jackie called this **livelock** or **divergence**: the system keeps executing transitions but makes no useful abstract progress. This is different from deadlock, where no event can proceed. The mechanism for ruling out such infinite new-event activity was left for the following lecture.

Key Terms, Notation, and Structures

Redundant guard
A candidate guard that is logically correct but already follows from the other guards, axioms, and pre-state invariants. Such a guard need not be stated explicitly in the event.

Complete before-after predicate
A relation between pre- and post-state variables that includes both changed and unchanged variables. For example, `IL_in` includes $c' = c$ even though c is not assigned by the event.

`skip`
An artificial dummy abstract transition used when a concrete new event has no abstract-level state change. In this session its effect is $n' = n$. It is not an actual M0 event.

New-event simulation
The refinement correspondence in which a transition of a new concrete event is matched by an abstract `skip`, while the refinement invariants hold across the corresponding pre- and post-states.

Livelock / divergence
A situation in which the system can keep executing, but the activity makes no useful abstract progress. In the demonstration, infinitely many concrete new events correspond to infinitely many abstract `skip`s.

Variant
The new concept Jackie previewed for the next lecture as the mechanism to be studied for preventing divergence. Its definition and proof obligations were not developed in Lecture 19.

Key Examples and Demonstrations

Example 1: Prove that $c = 0$ is redundant for `IL_in`

Setup. Assume the axioms, all pre-state invariants, and the necessary guard $a > 0$. The safety invariant contains

$$a = 0 \vee c = 0.$$

Main point. Since $a > 0$, the $a = 0$ disjunct cannot hold, so the model forces $c = 0$. The candidate guard is therefore provable rather than an independent restriction.

What to practise. Start from the sequent described on Jackie's annotated iPad notes p. 2. Use monotonicity to remove irrelevant hypotheses and construct the proof independently.

Example 2: Contrast the required $a = 0$ guard for `IL_out`

Setup. Assume the axioms, the pre-state invariants, and $b > 0$. Ask whether these conditions entail $a = 0$.

Main point. Jackie stated that the corresponding sequent is not provable. Therefore $a = 0$ is not redundant and must remain in the event guard.

What to practise. Explain why the two superficially symmetric guard questions have different results. Do not infer redundancy merely from the one-way-bridge intuition.

Example 3: Derive `skip` from the linking invariant

Setup. Use $a + b + c = n$ before and after `IL_in` or `IL_out`.

Main point. Each event transfers one car between two concrete counters, so one -1 and one $+1$ cancel in the total. Hence the abstract total is unchanged:

$$n' = n.$$

What to practise. Reproduce this reasoning for both new events without looking at Jackie's annotations, and explain why the result identifies the appropriate abstract dummy transition.

Example 4: Instantiate `IL_in/inv1_4/INV`

Setup. Begin with the complete hypotheses shown in Jackie's annotated iPad notes p. 5 and the post-state linking invariant.

Main point. Substitute the concrete effect of `IL_in` and the abstract effect of `skip` to obtain

$$(a - 1) + (b + 1) + c = n.$$

What to practise. Simplify the goal first, decide which hypotheses are actually relevant, and then discharge the proof using the inference rules referenced with the exercise. The notes pp. 6–7 provide sequent/rule references rather than a substitute for doing the proof yourself.

Example 5: Translate a concrete trace into an abstract trace

Setup. Follow one car from mainland to bridge, onto the island, back onto the bridge, and then to the mainland.

Main point. Old concrete events correspond to their old abstract events, while each new event contributes one abstract `skip`.

What to practise. Reconstruct both traces from Jackie's annotated iPad notes p. 8 and justify each event-to-event correspondence rather than memorizing the sequence.

Example 6: Why infinitely many `skip`s are a problem

Setup. In the deliberately weakened model on Jackie's annotated iPad notes p. 9, remove the guards from `IL_in` and `IL_out`.

Main point. The concrete model can repeatedly execute new events while the abstract model repeatedly performs `skip`. The system remains active but fails to make useful abstract progress: livelock/divergence.

What to practise. Explain why this is different from deadlock and why a finite number of `skip`s in an ordinary trace is not itself a problem.

Common Pitfalls and Misconceptions

- **Keeping every intuitively sensible guard.** A guard can be correct yet redundant. Check whether it follows from the existing guards, axioms, and pre-state invariants.
- **Assuming the two new-event guard questions are symmetric.** For `IL_in`, $c = 0$ is provable from the other conditions; for `IL_out`, $a = 0$ is not provable from $b > 0$ and the model constraints, so it must remain explicit.
- **Writing only changed variables in a BAP.** If an action omits a variable, its value is unchanged. Include the corresponding equality such as $c' = c$ or $a' = a$ when constructing the complete predicate.
- **Treating `skip` as an ordinary event added to M0.** Jackie introduced it as an artificial abstract transition for refinement reasoning. The original abstract model was not extended with a real `skip` event.
- **Forgetting the abstract effect when proving a linking invariant.** For a new event, the concrete variables change while the dummy abstract transition gives $n' = n$. Both sides matter when constructing the post-state linking-invariant goal.
- **Thinking every occurrence of `skip` is bad.** A finite number of `skip`s corresponding to a finite number of concrete new events is expected. The problem introduced here is an *infinite* sequence of new events/`skip`s.
- **Confusing livelock with deadlock.** Deadlock means no enabled event can make a transition. Livelock/divergence means transitions continue, but they make no useful abstract progress.

What You Should Be Able to Do

After this session, you should be able to:

- state the final guards and actions of `IL_in` and `IL_out`;
- formulate the proof question used to decide whether an additional guard is redundant, and explain why $c = 0$ is redundant for `IL_in` whereas $a = 0$ must remain for `IL_out`;
- write complete before-after predicates for the new-event actions, including unchanged variables;
- use the gluing invariant $a + b + c = n$ to show that either new event leaves the abstract value n unchanged;
- explain why the artificial abstract correspondence for a new event is `skip` with $n' = n$, and distinguish this reasoning device from an actual M0 event;
- count the invariant-preservation obligations associated with the two new events and construct a sequent such as `IL_in/inv1_4/INV`;
- translate a concrete event trace into the corresponding abstract trace by replacing new events with `skip`;
- distinguish deadlock from livelock/divergence and explain why an infinite sequence of new events corresponds to problematic infinite abstract skipping.

Suggested Practice

1. **Guard redundancy.** Starting from the axioms, all M1 pre-state invariants, and $a > 0$, prove that $c = 0$. Then compare this with the analogous attempt to derive $a = 0$ from the model constraints and $b > 0$.
2. **Before-after predicates.** Write the complete BAP for each new event without looking at Jackie's notes. Check explicitly which variable is unchanged in each event.
3. **Derive the dummy abstract effect.** Apply the two new-event BAPs to $a + b + c = n$ and show independently that $n' = n$ in each case.
4. **Invariant preservation.** Reconstruct `IL_in/inv1_4/INV`, simplify the goal, and discharge it. First use the displayed inference-rule references; later, closer to the exam, hide those references and try the proof again, as Jackie recommended.
5. **Trace correspondence.** Reproduce the one-car concrete trace and its abstract trace from memory. Explain why exactly two `skip`s appear.
6. **Enabledness checkpoint.** In the original guarded M1, revisit the state reached after the relevant `IL_in/IL_out` portion of the trace and verify Jackie's exercise that the new events are disabled there before comparing with the deliberately weakened model.

7. **Livelock diagnosis.** For the modified demonstration model, explain how removing the new-event guards permits an unbounded `IL_in/IL_out` cycle and why its abstract image is an infinite sequence of `skip`s.

Where to Review

- **Jackie's annotated iPad notes p. 2:** final guarded actions of `IL_in` and `IL_out`, including the redundant-versus-needed guard discussion and the linking/safety invariants used to justify it.
- **Jackie's annotated iPad notes p. 3:** complete before-after predicates for the new-event actions and the concrete-state picture.
- **Jackie's annotated iPad notes p. 4:** the commuting-diagram visualization for a concrete new event simulated by the artificial abstract `skip` transition.
- **Jackie's annotated iPad notes p. 5:** the invariant-preservation rule instantiated for a new event, the 2×5 counting question, and the `IL_in/inv1_4/INV` setup.
- **Jackie's annotated iPad notes pp. 6–7:** sequent and inference-rule references used for invariant-preservation proof practice.
- **Jackie's annotated iPad notes p. 8:** concrete versus abstract event traces and the finite `skip` correspondence.
- **Jackie's annotated iPad notes p. 9:** the deliberately modified M1 example that produces livelock/divergence through unbounded new-event execution.
- **Session recording:** return to Jackie's spoken explanation of how to test guard redundancy, why `skip` is only an artificial abstract correspondence, and the distinction between finite skipping and divergence.

Closing Reminder

Use this digest as a roadmap for active review, not as a substitute for the learning process itself. Attend Jackie's in-person sessions and pay attention to how he develops ideas and works through the important reasoning step by step. If a transition, explanation, example, or reasoning step remains unclear afterwards, return to the recording to fill that gap.

For structured review, start with Jackie's annotated iPad notes because they most closely reflect the actual flow and development of the class. Work through them in manageable logical units. After each logical unit, study the corresponding part of the presentation slides alongside it. Jackie does not normally teach by reading the slides line by line, so the slides may contain relevant supporting material that was not explicitly read aloud.

Finally, choose a suitable selection of the important reasoning from this session and work through it again on your own. The goal is not to memorize Jackie's recorded steps, but to determine whether you can reproduce the reasoning from your own understanding and explain why the important steps work.