UNIVERSITY OF CAMBRIDGE

Hoare Logic and Model Checking

Model Checking Lecture 4 Supplement

Conrad Watt

Computer Laboratory, University of Cambridge, UK
http://www.cl.cam.ac.uk/~caw77

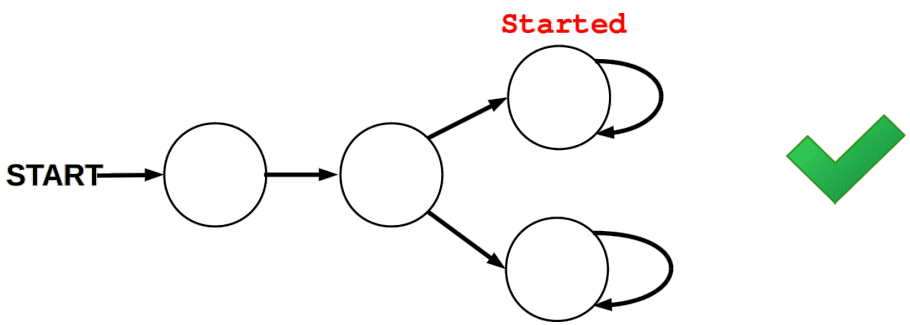
CST Part II – 2018/19

CTL formulas and models

- Examples from slide 91.
- Based on board-work during lecture 4.
- Example models, indicating whether the formula holds.
- Exercise: for failing models, give a counter-example path/trace.

CTL formulas and models (1)

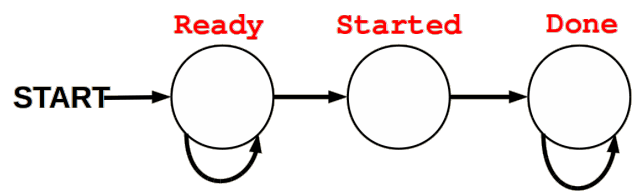
- "It is possible to get to a state where `Started` holds but `Ready` does not hold."
- **EF** (`Started` $\wedge$ $\neg$`Ready`)



- Exercise: compare to the **LTL** formula **F** (`Started` $\wedge$ $\neg$`Ready`)

CTL formulas and models (1)

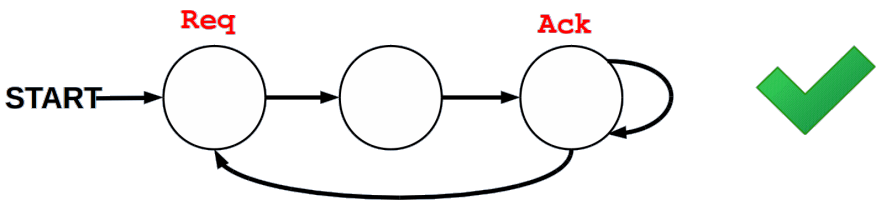
- "It is possible to get to a state where `Started` holds but `Ready` does not hold."
- **EF** (`Started` $\wedge$ $\neg$`Ready`)



- Exercise: compare to the **LTL** formula **F** (`Started` $\wedge$ $\neg$`Ready`)

CTL formulas and models (2)

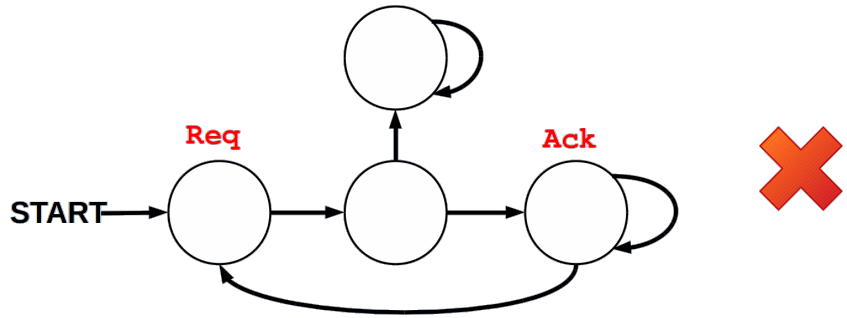
- "If a request `Req` occurs, then it will be eventually acknowledged by `Ack`."
- **AG** (`Req` $\Rightarrow$ **AF** `Ack`)



- Exercise: compare to the **LTL** formula **G** (`Req` $\Rightarrow$ **F** `Ack`)

CTL formulas and models (2)

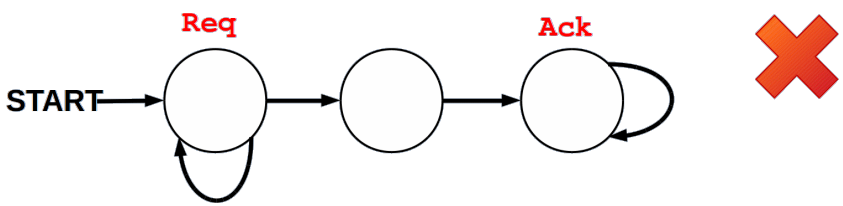
- "If a request `Req` occurs, then it will be eventually acknowledged by `Ack`."
- **AG** (`Req` $\Rightarrow$ **AF** `Ack`)



- Exercise: compare to the **LTL** formula **G** (`Req` $\Rightarrow$ **F** `Ack`)

CTL formulas and models (2)

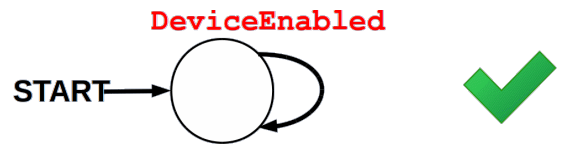
- "If a request `Req` occurs, then it will be eventually acknowledged by `Ack`."
- **AG** (`Req` $\Rightarrow$ **AF** `Ack`)



- Exercise: compare to the **LTL** formula **G** (`Req` $\Rightarrow$ **F** `Ack`)

CTL formulas and models (3)

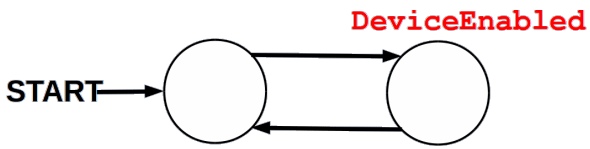
- "`DeviceEnabled` is always true somewhere along every path starting anywhere: i.e. `DeviceEnabled` holds infinitely often along every path."
- **AG** (**AF** `DeviceEnabled`)



- Exercise: compare to the **LTL** formula **G** (**F** `DeviceEnabled`)

CTL formulas and models (3)

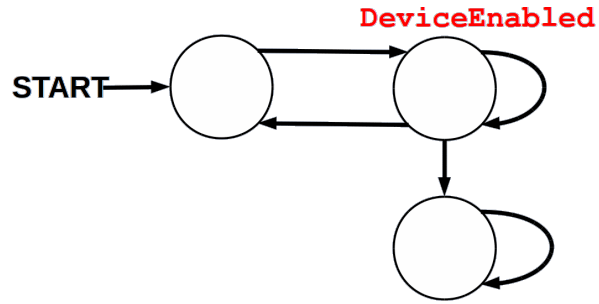
- "`DeviceEnabled` is always true somewhere along every path starting anywhere: i.e. `DeviceEnabled` holds infinitely often along every path."
- **AG** (**AF** `DeviceEnabled`)



- Exercise: compare to the **LTL** formula **G** (**F** `DeviceEnabled`)

CTL formulas and models (3)

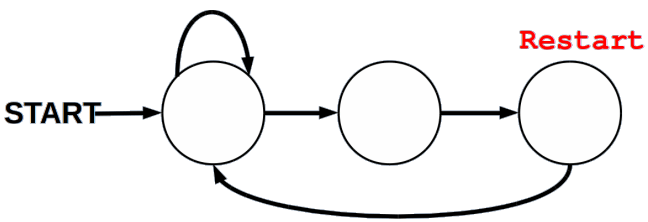
- "`DeviceEnabled` is always true somewhere along every path starting anywhere: i.e. `DeviceEnabled` holds infinitely often along every path."
- **AG** (**AF** `DeviceEnabled`)



- Exercise: compare to the **LTL** formula **G** (**F** `DeviceEnabled`)

CTL formulas and models (4)

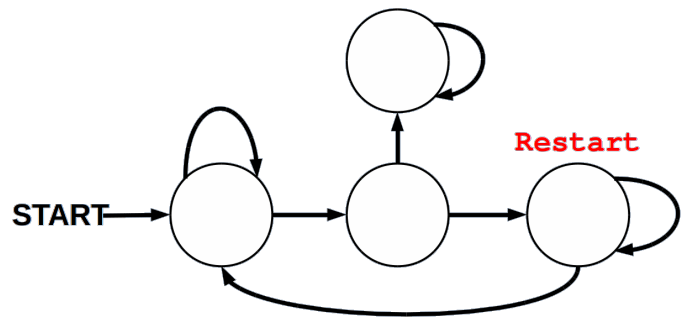
- From any state it is possible to get to a state for which "`Restart` holds."
- **AG** (**EF** `Restart`)



- Exercise: compare to the **LTL** formula **G** (**F** `Restart`)

CTL formulas and models (4)

- From any state it is possible to get to a state for which "`Restart` holds."
- **AG** (**EF** `Restart`)



- Exercise: compare to the **LTL** formula **G** (**F** `Restart`)

Misc CTL exercises (1)

- **AG** (Req ⇒ **AX**(**A**[¬Req **U** Ack]))
- Is the formula **AG** (Req ⇒ **A**[¬Req **U** Ack]) equivalent?
- Easy to construct a counter-example: the second formula requires that Ack is true immediately when Req is true.

Misc CTL exercises (2)

- **AG** (Req ⇒ (¬Ack ⇒ **AX**(**A**[Req **U** Ack])))
- Can we simplify the formula?

AG (Req ⇒ (¬Ack ⇒ **AX**(**A**[Req **U** Ack])))
≡ **AG** ((Req ∧¬Ack) ⇒ **AX**(**A**[Req **U** Ack]))
≡ **AG** ((Req ∧¬Ack) ⇒ (**A**[Req **U** Ack]))

- Exercise: are these equivalence steps correct?
  - Extended: do we have to assume that our model is left-total?