CS 512, Spring 2018, Handout 12

Dining Philosophers Problem

Assaf Kfoury

15 February 2018

The Dining Philosophers Problem (credit to Edsger Dijkstra)

(Graphic is courtesy of Benjamin D. Esham, depicting: Plato, Confucius, Socrates, Voltaire, Descartes)

- **Constraints**:
  1. Every philosopher eats or thinks, but not both.
  2. Every philosopher eats with two works, not with one only.
  3. When a philosopher stops eating, he puts both forks down (in sequence or not).
  4. There is a fixed eating-time interval for all philosophers.
  5. A fork can be used by only one philosopher at a time.
- **Problem**: Design a "protocol" such that none of the philosophers will starve, *i.e.*, they will forever alternate between eating and thinking.

The Dining Philosophers Problem (credit to Edsger Dijkstra)

(Graphic is courtesy of Benjamin D. Esham, depicting: Plato, Confucius, Socrates, Voltaire, Descartes)

- **Proposed protocol**:
  1. Every philosopher picks the left fork as soon as available.
  2. Every philosopher picks the right fork as soon as available.
  3. When a philosopher stops eating, he puts both forks down (in sequence, right then left).
  4. Repeat from step 1.
- **Proposed protocol is bad!** If all 5 philosophers pick up their left fork at the same time, they deadlock.
- **Question**: Is there a start state$^{(*)}$ for which there is a deadlock-free execution? (*) Necessarily requiring that not all 5 philosophers pick up the left fork at the same time.

The Dining Philosophers Problem (credit to Edsger Dijkstra)

Some properties of DPP expressed in CTL:

For $i = 1, \ldots, 5$, define the propositional atoms:

$e_i =$ philosopher i is eating,

$f_i =$ philosopher i has just finished eating.

1. $\forall\Box\ \neg(\, e_1 \wedge e_4)$, which says:

   "Philosophers 1 and 4 will never eat at the same time."

2. $\forall\,(\neg(\, e_1 \vee e_3 \vee e_4 \vee e_5\,)\ \ \mathbb{U}\ \ e_2)$ which says:

   "Philosopher 2 will be the first to eat."

3. $\forall\Box\ (\forall\Diamond\, e_1 \wedge \forall\Diamond\, e_2 \wedge \forall\Diamond\, e_3 \wedge \forall\Diamond\, e_4 \wedge \forall\Diamond\, e_5)$ which says:

   "Always every philosopher will get infinitely many turns to eat."

4. $\forall\Box\ (\,f_4 \rightarrow \forall(\,\neg e_4\ \mathsf{W}\ e_3)\,)$ which says:

   "Whenever philosopher 4 has finished eating, he cannot eat again until philosopher 3 has eaten."

The Dining Philosophers Problem (credit to Edsger Dijkstra)

Problems:

1. Design a transition system as a model of DPP.
2. Introduce the Randomized DPP
   (*Randomized Dining Philosophers Problem*) as follows:
   2.1 Every philosopher tosses a fair coin to decide which fork (left or right) to pick up first.
   2.2 If the fork chosen by the coin tossing is not available, then the philosopher repeats the random choice.
   2.3 If the fork chosen by the coin tossing is available, then the philosopher takes it and tries to get the other fork.
   2.4 If the other fork is not available, then the philosopher returns the taken fork and repeats from step 2.1.
3. Design a probabilistic transition system to model Randomized DPP.
4. Show that Randomized DPP is deadlock-free.

(THIS PAGE INTENTIONALLY LEFT BLANK)