Disjoint Conditions

CS 536: Science of Programming, Fall 2022

A. Why

- Disjoint parallel programs ensure that no thread can interfere with the execution of another thread.
- Disjoint conditions ensure that no thread can interfere with the conditions of a triple.
- Disjoint parallel programs with disjoint conditions can be proved correct by combining the proofs of their individual threads.

B. Objectives

At the end of this work you should be able to

- Recognize disjoint parallel programs and correctness triples with disjoint conditions
- Use the rules for sequentialization and disjoint parallelism

C. Questions

1. Suppose that $\{p_1\}\ S_1\ \{q_1\}$ and $\{p_2\}\ S_2\ \{q_2\}$ are parallel disjoint with disjoint conditions and let triple T_1* be $\{p_1 \wedge p_2\}\ [S_1 \parallel S_2]\ \{q_1 \wedge q_2\}$. Similarly, suppose $\{r_1\}\ S_1\ \{s_1\}$ and $\{r_2\}\ S_2\ \{s_2\}$ are a second pair of triples and let triple T_2* be $\{r_1 \wedge r_2\}\ [S_1 \parallel S_2]\ \{s_1 \wedge s_2\}$*. (Assume S_1 and S_2 are deterministic.)

   For all the parts below, include a brief explanation of why or why not.

   a. We know T_1* is a DPP (disjoint parallel program) with DC (disjoint conditions). Do we know anything about T_2* being a DPP?

   b. Do we know anything about T_2* having DCs?

   Say σ is a state that satisfies $p_1 \wedge p_2 \wedge r_1 \wedge r_2$ and say τ_1 is a final state reached when we run T_1* in σ. Similarly, let τ_2 be a final state reached when we run T_2* in σ.

   Let τ_1 be and let t_2 be a final state reached when we run T_2* in σ.

   c. Do we know if τ_1 and τ_2 are unique? Are they related and if so, how?

---

* The * in T_1* and T_2* indicate that T_1 and T_2 are partly or fully annotated with conditions.

2. Are the following programs parallel disjoint with disjoint conditions?
   - {T} x := 1 ; y := 1 {x = 1}
   - {x = 0} z := 0 {x = z}

i	***j***	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***
1	2					
2	1					

3. Are the following programs parallel disjoint with disjoint conditions?
   - {T} x := 1 ; y := 0 {x = 1}
   - {z = 0} z := z*x {z = 0}

i	***j***	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***
1	2					
2	1					

4. Are the following programs parallel disjoint with disjoint conditions?
   - {T} ***if*** x > 0 ***then*** y := 1; z := 2 ***fi*** {x ≤ 0 → z = 2}
   - {T} ***if*** x ≤ 0 ***then*** z := 2; y := 3 ***fi*** {x ≤ 0 → y = 3}

i	***j***	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***

5. Are the following programs parallel disjoint with disjoint conditions?
   - {T} x := u ; y := u {x = y}
   - {z > 0} z := z–1 ; v := z {v = z}
   - {w ≥ u} w := w+1 {w > u}

i	*j*	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***
1	2					
1	3					
2	1					
2	3					
3	1					
3	2					

6. Design a parallel program $\{p_1 \wedge p_2\}\ [\{p_1\}\ S_1\ \{q_1\} \parallel \{p_2\}\ S_2\ \{q_2\}]\ \{q_1 \wedge q_2\}$ where (1) the threads are not disjoint programs, (2) the threads don't have disjoint conditions, but (3) in fact, running the program in any state works correctly. There are any number of possible answers — it's easiest if you write the threads so that each one produces the same results whether you run it sequentially or in parallel with the other thread.

7. What do you get if you expand outlines (a), (b), and (d) from ***Example 6*** to get full outlines?

Solution to Practice 24

1. (Properties of disjoint parallelism and conditions)

1a. Whether or not triples include DPP's is a property of the programs in the triples; it's not related to the pre/post-conditions of the triples. The program $[S_1 \parallel S_2]$. in T_1* is a DPP, and T_2* has exactly the same program, so T_2* also is a DPP.

1b. We know T_1* has disjoint (pre/post)-conditions, but T_2* can have completely different conditions, so T_2* might have DCs or it might not.

1c. Since T_1* and T_2* have the same program, $[S_1 \parallel S_2]$, they have exactly the same set of final states, and by the unique result theorem for disjoint parallel programs, that result is unique. So $\tau_1 = \tau_2$ and they are unique.

Feel free to write {x, y} as x, y or just x y (which is what I used below.)

2. No: Thread 1 interferes with the conditions of thread 2

i	***j***	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***
1	2	x y	z	x z	Yes	No (because of) x
2	1	z	x y	x	Yes	Yes

3. No: Thread 1 interferes with the program of thread 2.

i	***j***	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***
1	*2*	x y	x z	z	No (because of) x	Yes
2	*1*	z	x y	x	Yes	Yes

4. No: Each interferes with the other's programs and conditions.

i	***j***	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***
1	2	y z	x y z	x y	No: y, z	No: y
2	1	y z	x y z	x z	No: y, z	No: z

5. Yes, these are parallel disjoint with disjoint conditions

i	***j***	***Change i***	***Vars j***	***Free j***	***Disjoint Program?***	***Disjoint Conditions?***
1	2	x y	v z	v z	Yes	Yes
1	3	x y	w	u w	Yes	Yes
2	1	v z	u x y	x y	Yes	Yes
2	3	v z	w	u w	Yes	Yes
3	1	w	u x y	x y	Yes	Yes
3	2	w	v z	v z	Yes	Yes

6. It helps if the threads change the state at opposite times. Here's one solution:

{T}
[{T} ***if*** x > 0 ***then*** y := 0 ***fi*** {x > 0 → y = 0)
|| {T} ***if*** x ≤ 0 ***then*** y := 0 ***fi*** {x ≤ 0 → y = 0}]
{(x > 0 → y = 0) ∧ (x ≤ 0 → y = 0)}
{y = 0}

More generally, if neither S_1 nor S_2 modify x, then the outline below is correct.

{T}
[{T} ***if*** x > 0 ***then*** S_1 ***fi*** {x > 0 → sp(x > 0, S_1)}
|| {T} ***if*** x ≤ 0 ***then*** S_2 ***fi*** {x ≤ 0 → sp(x ≤ 0, S_2)}]
{(x > 0 → sp(x > 0, S_1)) ∧ (x ≤ 0 → sp(x ≤ 0, S_2))}

7. Below are one result of expanding outlines (a), (b), and (d) from Example 6 to get full outlines. (There can be more than one right answer.)

   a. {x ≥ 0 ∧ y ≤ 0}
   [{x ≥ 0} z := x {z ≥ 0}
   || {y ≤ 0} w := –y {y ≤ 0 ∧ w = –y} {w ≥ 0}]
   {z ≥ 0 ∧ w ≥ 0}

   b. {z = 0}
   [{z = 0} x := z+1 {z = 0 ∧ x = z+1} {x ≤ z = 0}
   || {z = 0} y := z {z = 0 ∧ y = 0}]
   {x ≤ z = 0 ∧ y = z = 0}
   {x ≤ y = z = 0}

d. $\{x = y = z = c\}$
$\{x = c \wedge y = c \wedge z = c\}$
$[\{x = c\}\ \{x^2 = c^2\}\ x := x^2\ \{x = c^2\}$
$\|\ \{y = c\}\ \{y^2 = c^2\}\ y := y^2\ \{y = c^2\}$
$\|\ \{z = c\}\ z := (z-d)*(z+d)\ \{z_0 = c \wedge z = (z_0-d)*(z_0+d)\}\ \{z = c^2-d^2\}$
$]\ \{x = c^2 \wedge y = c^2 \wedge z = c^2-d^2\}$
$\{x = y = z+d^2\}$

