**Jacob Knispel & Alec Tiefenthal**

**Milestone 1 – Checking RDT 1.0**

4/28/2016

---

In this report, we will cover the accuracy of the two provided properties of the RDT1.0 protocol we modeled. It’s important to note that in order for sending and receiving to finish, it will take exactly *3d + 1* states where d is the number of data pieces being transferred. This is because, for a single piece of data to be transferred, the model progresses through 3 states:

State 1: Data is packeted in the sender

State 2: Data is sent to the receiver, still packeted

State 3: The receiver unpackets the data

The extra +1 comes from the initial state. This being said, however, our model can have *more than* 3d + 1 states, states will simply cease to change once the receiver has received all of the data the sender had originally.

*Property One: It is possible to transmit all of the data in the sender’s buffer to the receiver’s buffer.*

We found this property to be true. Below you’ll find an example of our trace, transmitAllData, running for just 4 states and 1 piece of data to be transferred for simplicity.

State 1:

This is the initial state. The sender’s buffer is full of all of the data that it plans to send to the receiver, and it has no packet yet. The receiver has no data and no packets.

State 2:

The sender has packeted the first piece of data, and so it now has a packet. Its buffer is missing {FILL\_IN}, since this is the piece of data that was packeted. The receiver has not changed.

State 3:

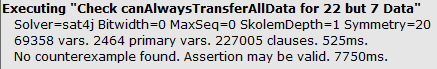
The sender has sent off its packeted piece of data, and so it no longer has it. The receiver now has a packeted piece of data, but its buffer remains empty.

State 4:

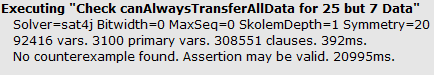
The receiver has unpacketed its piece of data. Its buffer now has the data that the packet contained, and the receiver has no packets. The sender remains unchanged.

*Property Two: Property One is always true*

We found this property to be true by checking an assertion that the final state is always an end state, found in our assertion canAlwaysTransferAllData. The screenshot below shows the assertion succeeding for 22 states and 7 pieces of data to be transferred.



Next, we show the assertion also holds true even when the number of states is not exactly equal to 3d + 1 by checking the assertion for 25 states and 7 pieces of data to be transferred.



*Conclusion*:

In summary, we found that, for this simple RDT1.0 model, data transfer is always possible and reliable given ample time to run to completion. This is likely because RDT1.0 ignores important factors like transfer time/accuracy.