Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

DEDUCING FAIRNESS PROPERTIES FOR UNITY PROGRAMS

Y.-K. Tsay
R. L. Bagrodia

September 1991
CSD-910069

Deducing Fairness Properties for UNITY Programs *

Yih-Kuen Tsay
yihkuen@cs.ucla.edu

Rajive L. Bagrodia
rajive@cs.ucla.edu

July 31, 1991

1 Introduction

Fairness is an important class of properties concerning programs whose semantics are modeled as sets of execution sequences. As each UNITY [CM88] program can be associated with a set of execution sequences, it is not unreasonable to talk about fairness properties of a UNITY program.

We provide three rules of inference that may be used to deduce fairness properties of the form "if p becomes *true* infinitely often then q will also become *true* infinitely often" (strong fairness) or "if p eventually remains *true* then q will become *true* infinitely often" (weak fairness) from a UNITY program. The degree of generality of these inference rules is illustrated by some of their special cases. We provide analogies to these rules that are expressible using the UNITY logic. The paper also contains an example that demonstrates the use of the inference rules in deriving a strong fairness property for a UNITY program.

2 Deducing Fairness

We adopt the linear temporal logic formula "$\Box\Diamond p$" to denote "p becomes *true* infinitely often" and "$\Diamond\Box p$" to denote "p eventually remains *true* (or p is eventually always *true*)." So, strong fairness can be expressed in the form of "$\Box\Diamond p \Rightarrow \Box\Diamond q$" and weak fairness in "$\Diamond\Box p \Rightarrow \Box\Diamond q$." *A UNITY program is said to have a fairness property if each of its execution sequence satisfies the corresponding formula.*

In the following presentation, we use the generic name M for an arbitrary program to denote a function from program states to a well-founded set under $\prec$. M may involve auxiliary variables.

The first inference rule is **Rule W-PROG** of [MP89] recast in the context of UNITY.

Theorem 1 $\dfrac{p \wedge (M = m) \mapsto (M \prec m) \vee q,\ (M \preceq m)\ \textit{unless}\ q}{\Box\Diamond p \Rightarrow \Box\Diamond q}$

Proof. Consider any execution sequence of a program that has the properties in the premise. Suppose p becomes *true* infinitely often in the execution sequence, but q never becomes *true* from i-th step. From $(M \preceq m)$ *unless* q, M will not increase from i-th step. From $p \wedge (M = m) \mapsto (M \prec m) \vee q$

*This research was partially supported by NSF under grant CCR 9157610 and by ONR under grand number N00014-91-J-1605.

and that q never becomes *true* from i-th step. M will decrease after each occurrence of p. Since there are infinitely many occurrences of p, M will keep decreasing from i-th step, which contradicts the assumption that M is a well-founded function. *End of Proof.*

We next show two special cases of theorem 1.

If M is defined such that $M = 1$ if q is *true* and $M = 0$ if q is *false*, we can have

$$\frac{p \mapsto q}{\Box\Diamond p \Rightarrow \Box\Diamond q}$$

Proof:

$p \wedge q \mapsto \neg q \vee q$	, implication theorem on $p \wedge q \Rightarrow \neg q \vee q$.	
$p \wedge (M = 1) \mapsto (M < 1) \vee q$	, from the above and the definition of M.	(1)
$p \wedge \neg q \mapsto p$	, implication theorem on $p \wedge \neg q \Rightarrow p$.	
$p \wedge \neg q \mapsto q$	, transitivity on the above and the premise $p \mapsto q$.	
$p \wedge (M = 0) \mapsto (M < 0) \vee q$	, from the above and the definition of M.	
$p \wedge (M = m) \mapsto (M < m) \vee q$	, from the above and (1).	(2)
$(M \leq 1)$ *unless* q	, from *true unless* q.	
$(M \leq 0)$ *unless* q	, from $\neg q$ *unless* q.	
$(M \leq m)$ *unless* q	, from the above two.	
The inference rule is valid	, from the above, (2), and theorem 1.	

If M is defined such that $M = 1$ if p is *true* and $M = 0$ if p is *false*, we can have

$$\frac{p \mapsto \neg p,\ \neg p \textit{ unless } q}{\Box\Diamond p \Rightarrow \Box\Diamond q}$$

Theorem 2 $\dfrac{\neg q \wedge (M = m) \mapsto q \vee (M \prec m) \vee \neg p,\ \neg p \textit{ unless } q}{\Box\Diamond p \Rightarrow \Box\Diamond q}$

Proof. Consider any execution sequence of a program that has the properties in the premise. Suppose p becomes *true* infinitely often in the execution sequence, but q remains *false* from i-th step. From $\neg p$ *unless* q and that q remains *false* from i-th step, p must remain *true* from i-th step; otherwise the occurrence of $\neg p$ followed by an occurrence of p will force q to become *true*. From $\neg q \wedge (M = m) \mapsto q \vee (M \prec m) \vee \neg p$ and that p remains *true* while q remains *false* from i-th step, M will keep decreasing from i-th step, which contradicts to that M is a well-founded function. *End of Proof.*

If M is defined such that $M = 1$ if q is *true* and $M = 0$ if q is *false*, we can have

$$\frac{\neg q \mapsto \neg p,\ \neg p \textit{ unless } q}{\Box\Diamond p \Rightarrow \Box\Diamond q}$$

Theorem 3 $\dfrac{p \mapsto \neg p \vee q}{\Diamond\Box p \Rightarrow \Box\Diamond q}$

Proof. Assume, from i-th step, p remains *true*. This trivially implies that p becomes *true* infinitely often. From the premise, q will become *true* after each occurrence of p from i-th step, which implies that q will become *true* infinitely often as p does. *End of Proof.*

3 Analogies in UNITY

The fairness properties described in the previous section cannot be expressed directly within the UNITY proof system. In this section, we present a set of three rules, each of which is analogous to the corresponding theorem in the previous section.

The closest analogy to theorem 1 in UNITY logic seems to be:

$$\frac{p \wedge (M = m) \mapsto (M \prec m) \vee q,\ (M \preceq m) \textit{ unless } q}{\text{Hypothesis: } \textit{true} \mapsto p \;\; \text{Conclusion: } \textit{true} \mapsto q}$$

(The hypothesis of the conditional property may be moved to the premise. The form of rule as presented is intended to reveal the similarity with theorem 1.)

From *true* $\mapsto p$ in F, we can deduce that at any point of the execution of program F, p will eventually become true, which implies that p becomes *true* infinitely often in each execution sequence of program F. So, the conditional property in the conclusion "roughly" says that if p becomes *true* infinitely often in each execution sequence, then q will also become *true* infinitely often in each execution sequence. Note that this property is weaker than the strong fairness property of theorem 1.

Proof:

$(M \preceq m)$ *unless* q	, from the premise.	
$\neg q$ *unless* q	, from the antireflexivity theorem of *unless*.	
$\neg q \wedge (M \preceq m)$ *unless* q	, simple conjunction on the above two.	
$p \wedge (M = m) \mapsto (M \prec m) \vee q$	, from the premise.	
$p \wedge \neg q \wedge (M = m) \mapsto (\neg q \wedge M \prec m) \vee q$	, PSP theorem on the above two.	(1)
$\neg q \wedge (M \prec m)$ *unless* q	, equivalent to $\neg q \wedge (M \preceq m)$ *unless* q.	
true $\mapsto p$	, from the hypothesis.	
$\neg q \wedge (M \prec m) \mapsto (p \wedge \neg q \wedge (M \prec m)) \vee q$	, PSP theorem on the above two.	(2)
$p \wedge \neg q \wedge (M = m) \mapsto (p \wedge \neg q \wedge (M \prec m)) \vee q$	, cancellation theorem on (1) and (2).	
$p \wedge \neg q \mapsto q$	, induction of the above.	(3)
$p \Rightarrow (p \wedge \neg q) \vee q$	, predicate calculus.	
$p \mapsto (p \wedge \neg q) \vee q$	, implication theorem on the above.	
$p \mapsto q$	, cancellation theorem on the above and (3).	
true $\mapsto q$	, transitivity on *true* $\mapsto p$ and the above.	

Similarly for theorem 2,

$$\frac{\neg q \wedge (M = m) \mapsto q \vee (M \prec m) \vee \neg p,\ \neg p \textit{ unless } q}{\text{Hypothesis: } \textit{true} \mapsto p \;\; \text{Conclusion: } \textit{true} \mapsto q}$$

Proof:

$\neg p \Rightarrow true$	, predicate calculus.
$\neg p \mapsto true$	, implication theorem on the above.
$true \mapsto p$	, from the hypothesis.
$\neg p \mapsto p$	, transitivity on the above two.
$\neg p$ *unless* q	, from the premise.
$\neg p \mapsto q$	, PSP theorem on the above two.
$\neg q \wedge (M = m) \mapsto q \vee (M \prec m) \vee \neg p$	, from the premise.
$\neg q \wedge (M = m) \mapsto (M \prec m) \vee q$	, cancellation theorem on the above two.
$(M \prec m) \vee q = (\neg q \wedge (M \prec m)) \vee q$	, predicate calculus.
$\neg q \wedge (M = m) \mapsto (\neg q \wedge (M \prec m)) \vee q$	, substitution axiom on the above two.
$\neg q \mapsto q$	, induction of the above.
$q \mapsto q$	, implication theorem on $q \Rightarrow q$.
$true \mapsto q$	, disjunction on the above two.

For theorem 3, we can have

$$\frac{p \mapsto \neg p \vee q}{\text{Hypothesis: } true \mapsto p \wedge b, \textbf{stable}\ p \wedge b \quad \text{Conclusion: } true \mapsto q}$$

where b is an auxiliary variable.

The hypothesis in the conclusion can deduce that p will eventually remain *true* [Mis90].
Proof:

$p \wedge b$ *unless false*	, from the hypothesis.
$p \mapsto \neg p \vee q$	, from the premise.
$p \wedge b \mapsto p \wedge b \wedge q$	, PSP theorem on the above two.
$p \wedge b \wedge q \mapsto q$	, implication theorem on $p \wedge b \wedge q \Rightarrow q$.
$p \wedge b \mapsto q$	, transitivity on the above two.
$true \mapsto p \wedge b$	, from the hypothesis.
$true \mapsto q$	, transitivity on the above two.

4 An Example

In the following program, a resource *res* periodically becomes available and a request *req* is also generated periodically but independently. When the resource becomes available, it remains available until it is consumed by some request (not necessarily the request represented by *req*). The request *req* will persist until it consumes the resource *res*. We desire to show that if some request *req* is generated and subsequently the requested resource *res* becomes available infinitely often, then eventually the request will be granted. The UNITY program is as follows:

```
Program $F$
    declare     $a,res$ : boolean,
                $b,req$ : boolean,
                $pri$ : integer
    initially   $a,res = false,false$
              [] $b,req = false,false$
              [] $pri = 0$
    assign      $a := \neg a$
              [] $res := true$              if $\neg res \wedge a$
              [] $b := \neg b$
              [] $req,pri := true,K$        if $\neg req \wedge b$ /* $K$ is some positive integer */
              [] $res := false$             if $res \wedge \neg req$
              [] $res,pri := false,pri - 1$ if $res \wedge req \wedge (pri > 0)$
              [] $res,req := false,false$   if $res \wedge req \wedge (pri = 0)$
end {$F$}
```

It can be shown that

invariant $0 \leq pri \leq K$ in F

$(req \wedge res) \wedge (pri = k) \mapsto (pri < k) \vee \neg req$ in F

$(pri \leq k)$ *unless* $\neg req$ in F

From theorem 1, program F has the property $\Box\Diamond(req \wedge res) \Rightarrow \Box\Diamond\neg req$. The request *req* will eventually be granted if the resource *res* is available infinitely often.

5 Conclusion

Three inference rules that may be used to deduce fairness properties were described. Although the fairness properties themselves cannot be expressed in the UNITY logic, the premise for each rule defines conditions expressible in the UNITY logic that are sufficient to deduce the corresponding fairness property. Such a premise may be used to indirectly specify strong fairness properties for UNITY programs.

Acknowledgment

We thank Jayadev Misra and Ambuj Singh for their comments on an early draft of this paper.

References

[CM88] K.M. Chandy and J. Misra. *Parallel Program Design: A Foundation.* Addison-Wesley, 1988.

[Mis90] J. Misra. Auxiliary variables. *Notes on UNITY: 10-89*, July 10 1990.

[MP89] Z. Manna and A. Pnueli. The anchored verson of the temporal framework. In J.W. de Bakker, W.P. de Roever, and Rozenberg G., editors, *LNCS 354: Linear Time, Branch-*

ing Time and Partial Order in Logic and Models for Concurrency, pages 201–284. Springer-Verlag, 1989.