Under consideration for publication in Math. Struct. in Comp. Science

Proving Liveness Property under a Mixture of Strengthened Compassion and Compassion Requirements†

TENG LONG[1,2] and WENHUI ZHANG[1]

[1] *State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, P.R.China.*
[2] *School of Information Science and Engineering, Graduate University of China Academy of Sciences, P.R.China.*

Received 12 October 2012

Liveness property is among the most important properties of programs. Many methodologies have been proposed for proving liveness properties. This paper studies deductive rules for proving liveness properties under different kinds of fairness requirements (constraints). It is based on method where a program is augmented by a non-constraining progress monitor based on a set of ranking functions, and further abstracted by predicate abstraction in order to allow the use of algorithmic verification techniques. The main contributions are a deductive rule under different fairness requirements (constraints) for proving liveness properties and an algorithm for automatically deriving deductive proof constructs. We show how the constructs necessary for deductive proofs of liveness properties can be automatically extracted.

1. Introduction

Liveness properties are requirements that something good must eventually happen. A counterexample to such a property is typically a loop during which the good thing never occurs. One frequently used subtype of liveness properties is response properties. It is an important and widely studied subtype of liveness properties (Pnueli and Sa'ar, 2008; Manna and Pnueli, 2010). Our work focuses on this type of liveness properties that has the form $p \Longrightarrow \diamond q$ (abbreviating $\Box(p \rightarrow \diamond q)$), where p and q are assertions.

The systems to be analyzed may be modeled as discrete systems (Kesten et al., 1998). For avoiding acceptance of unrealistic loops in such models, e.g., in which some process or action is infinitely ignored, fairness is needed for imposing restrictions on accepted runs on such models. For dealing with different situations, several different notions of fairness have been proposed. In 1981, Lehmann, Pnueli and Stavi defined *justice* requirements (Lehmann et al., 1981) to describe the situation that some states must be visited infinitely

† Supported by the National Natural Science Foundation of China under Grant Nos. 60833001, 61272135, and the CAS Innovation Program. Emails: {longteng, zwh}@ios.ac.cn. Corresponding author: Teng Long.

often. *Compassion* is a kind of generalization of justice, suggested by Pnueli and Sa'ar (Pnueli and Sa'ar, 2008), and may be represented by a set of pairs of assertions of the form $\langle \psi, \varphi \rangle$. This fairness condition requires that along every path, for each pair $\langle \psi, \varphi \rangle$, either ψ is true only finitely many times or φ is true infinitely often.

In this paper, we study a kind of fairness named *strengthened compassion*, i.e., compassion with some additional constraints, represented by a set of one-step pairs of assertions. An important characteristics of this kind of fairness is the constraint of transition (involving states and their successors).

For the verification of response properties of systems with strengthened compassion constraints and a mixture of others, we present a framework based on ranking abstraction (Kesten and Pnueli, 2000; Balaban et al., 2005; Balaban et al., 2007) and deductive rules. Ranking abstraction is based on an augmentation of the concrete program. The augmentation is parameterized by a set of well-founded ranking functions. Based on these, new fairness requirements are generated, all of which are synchronously composed with the program in a non-constraining manner. Ranking functions in this approach are not expected to decrease with each transition of the program. The system that satisfies the response property is guaranteed to leave non-decrease transitions (loops) by fairness requirements.

Then since for the use of the deductive rules for proving response properties, a set of auxiliary constructs, such as ranking functions and helpful assertions, is needed, we present an algorithm for deriving such auxiliary constructs. Previously, invariants as a boolean combinations of the given predicates are derived by predicate abstraction(Graf and Saïdi, 1997; Ball et al., 2001) for proving safety properties (Manna and Pnueli, 1991), and well-founded ranking functions and assertions are extracted from a successful application of the ranking abstraction method for a deductive proof of response properties (Pnueli and Sa'ar, 2008).

The contribution of the paper is as follows: it suggests a formal framework, based on abstraction methods, for proving liveness properties, including deductive rules under different fairness, and methods for automatically deriving a global well-founded ranking function.

The paper is organized as follows: In section 2, the definitions of different kinds of fairness requirements and the computational model of fair discrete systems under strengthened compassion are introduced. Section 3 proposes the deductive rule under strengthened compassion and proves the soundness and relative completeness of the rule. It also presents a method for extracting the auxiliary constructs necessary for a deductive proof under strengthened compassion of a response property. These auxiliary constructs include a set of ranking functions and helpful assertions. The application of the method is illustrated by an example in section 4. In section 5, the deductive rule and the method are extended to deal with systems with both compassion and strengthened compassion requirements. Finally, section 6 contains concluding remarks.

Related works: There have been several works dealing with the extraction of proofs from successful application of a model checking run. The papers (Peled and Zuck, 2001) and (Peled et al., 2001) show how to construct a deductive proof for verifying linear temporal logic formulas. In (Namjoshi, 2001), a certificate that confirms the correctness

of a successful model checking run, can be viewed as a deductive proof of the verified property, where the proof is presented as an automaton similar to the verification diagrams of (Manna and Pnueli, 1994). In (Kupferman and Vardi, 2005), it shows how to construct an automaton certificate that confirms the correctness of the verified property, and can be checked automatically and efficiently.

The above methods were applied to propositional properties of finite-state systems and produced proofs (or certificates) that were propositional in nature. None of them involved a ranking function over an unbounded domain. In (Namjoshi, 2003), the author translated a proof of a finite-state abstraction of an infinite-state system by referring to the proof concretization process as lifting of the proof. The limitation of the method was that the lifting of ranking functions necessarily preserve the range of the functions. Starting from a finite-state system, the range of the abstract ranking functions and, therefore, the resulting range of the concrete ranking functions is necessarily bounded. In comparison, the methods presented here in Sections 3 and 5 extract ranking functions over unbounded domains from their finitary representation as fairness requirements.

There has been a lot of research work on fairness. In (Sun et al., 2009), a process analysis toolkit for system analysis with different kinds of fairness was presented. They took *fairness constraints* into verification algorithms instead of re-formulating the property. It considered finite-state systems only. For dealing with infinite-state system, (Kesten and Pnueli, 2000) proposed an augmented finitary abstraction to verify properties. In (Balaban et al., 2007), a method based on analysis of maximal strongly connected components by ranking abstraction for proving safety and liveness properties was presented. It can deal with the system under justice and additional compassion requirements (include additional variable *dec*) introduced by ranking abstraction. To complement the absence of the native compassion requirements (i.e., the compassion requirements without the additional variable *dec* that comes from ranking abstraction), (Long and Zhang, 2010) extended the method to deal with compassion requirements that are considered as one of the components in a system. (Balaban et al., 2010) presented an automatic method to derive deductive proofs of liveness properties from symbolic model checking under compassion.

This paper is an extension of the work presented in TAMC 2012 (Long and Zhang, 2012). The extension includes extending the rule and the method for proving response properties to deal with systems under a mixture of both strengthened compassion and compassion constraints.

2. Fair Discrete Systems under Strengthened Compassion

As explained in the introduction, there are several different notions of fairness for dealing with different situations. We consider strengthened compassion that is a kind of fairness based on compassion with additional constraints.

Let V be a set of variables. An assignment of values to the variables of V is called a state. A state formula (also called an assertion) is a formula φ over V representing the set of states that satisfies φ. A state that satisfies φ is called a φ-state.

A strengthened compassion requirement is a set of pairs of assertions denoted as follows.

$$\mathcal{SC} = \{\langle r_i, \{u_i\}\rangle \mid i = 1, ..., n\}.$$

An infinite sequence of states $\sigma = s_0 s_1 \cdots$ satisfies the strengthened compassion requirement $\mathcal{SC}$, if for each $i \in \{1, ..., n\}$, σ contains only finitely many r_i-states, or σ contains infinitely many r_i-states and if s_m is such a state (i.e., $s_m \models r_i$), then $s_{m+1} \models u_i$.

2.1. *Strengthened Compassion Discrete Systems*

A strengthened compassion discrete system is similar to a fair Kripke structure defined in (Kesten et al., 1998). The difference is that the fairness requirements are different.

Definition 2.1. A strengthened compassion discrete system (SCDS) is a quadruple $\mathcal{D} = \langle V, \Theta, \rho, \mathcal{SC}\rangle$ where the components are as follows.

— V : A finite set of typed *system variables* - containing data and control variables. A set of states (interpretation) over V is denoted by Σ. For a state s and a system variable $v \in V$, we denote by $s[v]$ the value assigned to v by the state s.

— Θ : The *initial condition* - an assertion (state formula) characterizing the initial states.

— ρ : The *transition relation* - an assertion $\rho(V, V')$, relating the values V of the variables in state $s \in \Sigma$ to the values V' in a $\mathcal{D}$-successor state $s' \in \Sigma$.
If φ is a formula representing a set of states, then φ' is the formula with each v replaced by v'.

— $\mathcal{SC}$: The *fairness requirements* - a set of one-step pairs of assertions $\mathcal{SC} = \{\langle r_i, \{u_i\}\rangle | i = 1, ..., n\}$.

Computation: A computation of $\mathcal{D}$ is an infinite sequence of states $\sigma = s_0 s_1 \ldots$, satisfying the following requirements:

— $s_0 \models \Theta$.

— For each $j = 0, 1, \ldots$, the state s_{j+1} is in a D-successor of the state s_j. For each $v \in V$, we interpret v as $s_j[v]$ and v' as $s_{j+1}[v]$, such that $\langle s_j, s_{j+1}\rangle \models \rho(V, V')$.

— σ satisfies the fairness requirements.

2.2. *A Discussion of Different Kinds of Fairness*

Justice is a simple kind of fairness that may be represented by a set of state formulas $\{\varphi_1, ..., \varphi_n\}$ and this fairness condition requires that each φ_i must be true infinitely often along every path. *Compassion* is a kind of generalization of justice, and may be represented by a set of pairs of state formulas $\{\langle\psi_1, \varphi_1\rangle, ..., \langle\psi_n, \varphi_n\rangle\}$. This fairness condition requires that along every path, for each pair $\langle\psi, \varphi\rangle$, either ψ is true only finitely many times or φ is true infinitely often. *Justice* and *Compassion* are regarded as weak and strong fairness in (Lamport, 1977), both of them constrain the fairness of actions. They can deal with actions fairly in *one context*. On the other hand, the correctness of many population protocols rely on stronger fairness constraints that may constrain actions in *all contexts*, e.g., self-stabilizing leader election in ring networks (Fischer and Jiang, 2006) and token circulation in rings (Angluin et al., 2005). *Strengthened Compassion* may deal

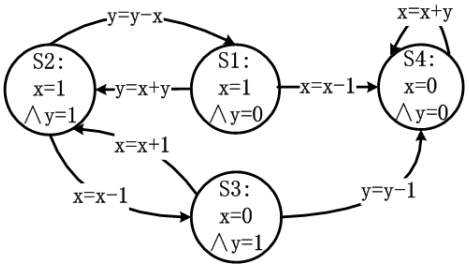

Fig. 1. Transitions for Comparison of Compassion and Strengthened Compassion

with such situations, and an important characteristics of strengthened compassion is the constraint of transition (involving states and their successors). Consider the transitions in Fig. 1, the differences between compassion and strengthened compassion are explained as follows.

— Compassion requirements constrains that infinitely enabled actions must eventually be taken.
The compassion requirement: $\langle x = 1 \wedge y = 0, x = 0\rangle$ constrains that if "$x = 1 \wedge y = 0$" is satisfied infinitely many times, then "x=0" must be satisfied infinitely many times. The infinite loop $(S_1S_2S_3S_2)^\omega$ satisfy this constraint. In such a loop, the action "x=x-1" is enabled infinitely many times, and this action is taken infinitely times.
The action "x=x-1" in the figure is enabled in both S_1 and S_2, however, the compassion requirement does not distinguish the two different transitions.

— Strengthened compassion requirements constrains that transitions with an infinitely enabled action must eventually be taken.
The strengthened compassion requirement $\langle x = 1 \wedge y = 0, \{x = 0\}\rangle$ constrains that if "$x = 1 \wedge y = 0$" is satisfied infinitely many times, then "x=0" must be satisfied infinitely many times *right after* "$x = 1 \wedge y = 0$".
The infinite loop $(S_1S_2S_3S_2)^\omega$ does not satisfy this requirement. In such a loop, the action "x=x-1" is enabled infinitely many times at S_1 and S_2, but the action "x=x-1" is never taken from S_1 (which satisfies $x = 1 \wedge y = 0$) in the loop (i.e., the transition $S_1 \xrightarrow{x=x-1} S_4$ is not taken).

2.3. *Soundness of the Rule*

The soundness is established as follows. Suppose that the premises of the rule are valid and the conclusion is not. We prove that this is a contradiction.

The conclusion is not valid means that there exists a computation $\sigma = s_0, s_1 \ldots$ and a position $j \geq 0$ such that $s_j \models p$ and no state s_k, for $k \geq j$ satisfies q. Without loss of generality, we take $j = 0$. According to premises of R_1 and R_2 and the assumptions that no states satisfy q, then any state s_w satisfies $r_i \wedge \varphi_i$ for some $i \in \{1, ..., n\}$. Since there are only finitely many different i's, there exists a cutoff index $h \geq 0$ such that for every i and $w \geq h$, $s_w \models r_i \wedge \varphi_i$ if and only if σ contains infinitely many $(r_i \wedge \varphi_i)$-positions.

Consider position $w_1 = h$. Choose i_1 to be the index such that $s_{w_1} \models r_{i1} \wedge \varphi_{i1}$.

Let p, q be assertions.
Let $\mathcal{A} = (W, \succ)$ be a well-founded domain.
Let $\mathcal{SC} = \{F_i \mid F_i = \langle r_i, \{u_i\}\rangle, i \in \{1, ..., n\}\}$ be a set of strengthened compassion requirements.
Let $\{\varphi_i \mid i \in \{1, ..., n\}\}$ be a set of assertions.
Let $\{\Delta_i : \Sigma \to \mathcal{A} \mid i \in \{1, ..., n\}\}$ be a set of ranking functions.

R1	p	$\Rightarrow$	$q \vee \bigvee_{j=1}^{n}(r_j \wedge \varphi_j)$
$\forall i \leq n$:			
R2	$r_i \wedge \varphi_i \wedge \rho$	$\Rightarrow$	$q' \vee \bigvee_{j=1}^{n}(r'_j \wedge \varphi'_j)$
R3	$\varphi_i \wedge \rho$	$\Rightarrow$	$q' \vee (\varphi'_i \wedge \Delta_i = \Delta'_i) \vee \bigvee_{j=1}^{n}(r'_j \wedge \varphi'_j \wedge \Delta_i \succ \Delta'_j)$
R4	$\varphi_i \wedge r_i \wedge \rho \wedge \varphi'_i$	$\Rightarrow$	$\neg u'_i$
	p	$\Rightarrow$	$\diamond q$

Fig. 2. Deductive Rule: SC_RESPONSE

According to R_3 and the assumption that σ contains no q-positions, then either φ_{i1} holds at all positions $w \geq w_1$, or there exists a position $w_2 \geq w_1$ and index i_2 such that $s_{w_2} \models r_{i2} \wedge \varphi_{i2}$ and $\Delta_{i1}(s_{w_1}) \succ \Delta_{i2}(s_{w_2})$. We argue that the former case is not possible and then the latter leads to an infinite sequence of decreasing values of Δ.

— If φ_{i1} holds continuously beyond w_1, then due to premise of R_4, $r_{i1} \wedge \varphi_{i1}$ holding at $w_1 \geq h$ implies that $r_{i1} \wedge \varphi_{i1}$ (and therefore r_{i1}) holds at infinitely many positions without succeed infinitely many u_{i1}-states. This violates the requirement $\langle r_{i1}, \{u_{i1}\}\rangle$.
— If there exists a position $w_2 \geq w_1$ and index i_2 such that $s_{w_2} \models r_{i2} \wedge \varphi_{i2}$ and $\Delta_{i1}(s_{w_1}) \succ \Delta_{i2}(s_{w_2})$, we can continuously find i_3, i_4, ..., such that $\Delta_{i_1}(s_{w_1}) \succ \Delta_{i_2}(s_{w_2}) \succ \Delta_{i_3}(s_{w_3}) \succ \ldots$. According to the definition of well-founded domain, it is impossible to find infinite positions to satisfy the decrease sequence.

Therefore there cannot exist a computation σ violating the response property $p \Rightarrow \diamond q$ if the premises of rule are all valid.

3. Proving Response Properties under Strengthened Compassion

For verifying response properties under the assumption of strengthened compassion requirements over an SCDS, the deductive rule SC_RESPONSE is presented in Fig. 2. The premisses of the rule are explained as follows.

For proving $p \Rightarrow \diamond q$, the rule deals with the pend states (all the reachable states from p-states though $\neg q$-path). Helpful assertions φ_i and ranking functions Δ_i where each $\Delta_i : \Sigma \to \mathcal{A}$ is defined on a well-founded domain $\mathcal{A}$ are the effective premises of the rule, and $(r_i \wedge \varphi_i)$-states are the states in φ_i constrained by fairness requirement $\langle r_i, u_i\rangle \in \mathcal{SC}$.

R1 requires that any p-state is either a goal state (i.e., a q-state), or a $(r_i \wedge \varphi_i)$-state for some $i \in \{1, \ldots, n\}$. R2 requires that any step from a $(r_i \wedge \varphi_i)$-state moves either directly to a q-state, or to another $(r_j \wedge \varphi_j)$-state, or stays at a state of the same level (i.e., a $(r_i \wedge \varphi_i)$-state). R3 requires that any step from a φ_i-state moves either directly to a q-state, or to another $(r_j \wedge \varphi_j)$-state with decreasing rank $(\Delta_i \succ \Delta_j)$, or stays at a

state with the same rank. R4 ensures that during the transitions among the states inside of φ_i, there are no transitions from r_i-states to u_i-states.

R_{2-4} together with the definition of strengthened compassion requirements guarantee that if an execution enters a loop (consisting of states of some φ_i) without leaving it, then there exists strengthened compassion requirements it violates. It must get out of all the unfair loops and finally reach q-state (the goal state).

3.1. *Soundness of the Rule*

The soundness is established as follows. Suppose that the premises of the rule are valid and the conclusion is not. We prove that this is a contradiction.

The conclusion is not valid means that there exists a computation $\sigma = s_0, s_1 \ldots$ and a position $j \geq 0$ such that $s_j \models p$ and no state s_k, for $k \geq j$ satisfies q. Without loss of generality, we take $j = 0$. According to premises of R_1 and R_2 and the assumptions that no states satisfy q, then any state s_w satisfies $r_i \wedge \varphi_i$ for some $i \in \{1, ..., n\}$. Since there are only finitely many different i's, there exists a cutoff index $h \geq 0$ such that for every i and $w \geq h$, $s_w \models r_i \wedge \varphi_i$ if and only if σ contains infinitely many $(r_i \wedge \varphi_i)$-positions.

Consider position $w_1 = h$. Choose i_1 to be the index such that $s_{w_1} \models r_{i1} \wedge \varphi_{i1}$. According to R_3 and the assumption that σ contains no q-positions, then either φ_{i1} holds at all positions $w \geq w_1$, or there exists a position $w_2 \geq w_1$ and index i_2 such that $s_{w_2} \models r_{i2} \wedge \varphi_{i2}$ and $\Delta_{i1}(s_{w_1}) \succ \Delta_{i2}(s_{w_2})$. We argue that the former case is not possible and then the latter leads to an infinite sequence of decreasing values of Δ.

— If φ_{i1} holds continuously beyond w_1, then due to premise of R_4, $r_{i1} \wedge \varphi_{i1}$ holding at $w_1 \geq h$ implies that $r_{i1} \wedge \varphi_{i1}$ (and therefore r_{i1}) holds at infinitely many positions without succeed infinitely many u_{i1}-states. This violates the requirement $\langle r_{i1}, \{u_{i1}\}\rangle$.
— If there exists a position $w_2 \geq w_1$ and index i_2 such that $s_{w_2} \models r_{i2} \wedge \varphi_{i2}$ and $\Delta_{i1}(s_{w_1}) \succ \Delta_{i2}(s_{w_2})$, we can continuously find i_3, i_4, ..., such that $\Delta_{i_1}(s_{w_1}) \succ \Delta_{i_2}(s_{w_2}) \succ \Delta_{i_3}(s_{w_3}) \succ \ldots$. According to the definition of well-founded domain, it is impossible to find infinite positions to satisfy the decrease sequence.

Therefore there cannot exist a computation σ violating the response property $p \Rightarrow \diamond q$ if the premises of rule are all valid.

3.2. *An Algorithm for the Construction of Auxiliary Constructs*

For the application of the rule SC_RESPONSE, it is essential that we can construct the auxiliary constructs used in the premisses of the rule, including ranking functions and assertions. In the following, we provide an algorithm for the construction of the auxiliary constructs. The algorithm works for finite state models.

3.2.1. *Algorithm SC_Auxiliary_Constructs* Considering an SCDS $\mathcal{D}$ and a response property $p \Rightarrow \diamond q$, we present an algorithm which extracts a deductive proof according to the rule of a response property $p \Rightarrow \diamond q$. It defines the values $\delta_1, \ldots, \delta_m$ of the respective ranking functions $\Delta_1, \ldots, \Delta_m$ on different sets of states, a helpful assertion φ_i and a

Algorithm 1 *SC_Auxiliary_Constructs*

1: $m := 0$
2: $accessible_{\mathcal{D}} := E(trueS\Theta)$
3: $pend := accessible_{\mathcal{D}} \wedge E(\neg qS(p \wedge \neg q))$
4: $rank\_SC(pend, [])$

where the procedure $rank\_SC$ is defined as follows.
procedure $rank\_SC(subpart,prefix)$
d:**integer**
Y:**assertion**
5: Let $d := 0$
6: Let $Y := subpart$
7: FIX (Y)
8: **Forall** $(\langle r, \{u\}\rangle \in SC)$ **do**
9: Let $\psi = Y \wedge \neg(Y \wedge r \wedge EX(Y \wedge u))$
10: **if** $\psi \wedge r \neq \emptyset$ **then**
11: Let $m = m + 1$
12: Let $d = d + 1$
13: Let $\varphi_m := E(\psi S(\psi \wedge r))$
14: Let $h_m := \langle r, \{u\}\rangle$
15: Let $\delta_m := prefix * [d]$
16: Let $Y := Y \wedge \neg\varphi_m$
17: Let $rem := \varphi_m \wedge \neg r$
18: **if** $(rem \neq \emptyset)$ **then**
19: $rank\_SC(rem, prefix * [d])$
20: **end if**
21: **end if**
22: end for
23: **if** $(Y \neq \emptyset)$ **then**
24: report "fail"
25: **end if**
26: **end-Fix**

fairness requirement $h_i = \langle r, \{u\}\rangle$, which can constraint φ_i for each $i \in \{1, .., m\}$. The algorithm *Auxiliary_constructs* is presented as Algorithm 1 and is explained as follows.

For assertions p and q, the formula $\mathbf{E}(p\,\mathcal{S}\,q)$ captures the set of states that are reachable from a q-state by a p-path all of whose states, except possibly the first, satisfy p. In this expression we use the *since* temporal operator $\mathcal{S}$. The formula $\mathbf{EX}(p)$ captures the set of states that are the immediate predecessors of p-states.

The expression $accessible_{\mathcal{D}}$ captures the set of all accessible states with $\mathcal{D}$. The expression of *pend* describes pend states: all states which are reachable from any accessible p-state by a finite q-free path. *prefix* is a list that is supposed to be a prefix of the value of ranking function. The list operation $*$ denotes the concatenation of two lists. ψ is the

set of Y-states without r-states which are the predecessors of u_j-states. φ is the set of ψ-states which can be reached by r, i.e. φ-states are those that form a strongly connected subgraph of ψ. *rem* is the set of φ-states that are not r-states. The new Y is the set of remaining states.

For each i, φ_i, h_i, δ_i where $\Delta_i(s) = \delta_i$ for $s \in \varphi_i$, are the auxiliary constructs discovered at the respective stages of the execution of the algorithm. For each strengthened compassion $h_i = \langle r, \{u\}\rangle$, we construct φ_i (the set of states that formed an unfair loop that contains r-states which are not the predecessor of u-states for some j) with its own δ_i to measure the distance between the loop to the goal states. The construction is as follows:

— S1: To start with, we deal with the *pend* states (line 4), i.e. $Y_0 = pend$. By the definition of *pend*, we know that there are no goal states in *pend*. The first unfair loop we can find, is the one nearest to goal states (under the strengthened compassion for the transition to goal states).
— S2: For each m, after removing an unfair loop φ_m, the new set of states we will be dealing with is $Y' = Y - \varphi_m$ (line 16). In Y', by calling $rank\_SC$ (line 19) recursively, we construct φ_{m+1} and δ_{m+1}.
— S3: According to the definition of strengthened compassion, the reason of unfairness is the "bad"states (the r-states) in the loop (line 10). Therefore we remove r-sates from each φ_i (line 17), and then we deal with the remaining part of φ_i recursively (line 18, 19). The ranks of states in φ_i are with the same prefix δ_i.
— S4: If all the *pend* states consist of unfair loops which can be constrained by strengthened compassion to leave the *pend* states, then the liveness property can be guaranteed. Otherwise, the liveness property fails (line 23, 24).

Additional explanation of the algorithm is as follows.

— Line 7: FIX Y terminates when Y does not change after the specified computation. After termination, it is at line 23 which prepares the final result for S4.
— Line 9: Constructing ψ by removing the r states that enable the transition $r \rightarrow u$ from Y. Then there are no transitions of the form $r \rightarrow u$ in ψ_i. Then if r is part of a loop in ψ, then this loop must be unfair violating the fairness requirement under consideration.
— Line 10: Checking whether there may exist such unfair loop by checking whether there exist r-states in ψ_i.
— Line 11: Constructing assertion φ which consists of all of the reachable states from r-states in ψ_i.
— Line 13-15: Constructing the helpful assertion φ_m, the strengthened compassion constraint h_m and the distance measure δ_m, respectively.
— Line 18-22: Constructing the new Y for the use by S2, and preparing the recursive call for S3.

3.2.2. *Validity of the Algorithm* For every Y at different levels of the recursive calls of the algorithm, if there exist a fairness constraint $\langle r, \{u\}\rangle$ such that there is no $\langle s_j, s_{j+1}\rangle \models \rho(V, V')$ such that $s_j \models r$ and $s_{j+1} \models u$, exists at least one r-state which is not the

predecessor of u-states, then this fairness constraint is sufficient to guarantee that it is not possible to stay at φ-states (the reachable states from r-states) infinitely often. Otherwise, if during all fairness constraints, there is no such j exist, the execution of the model are not required to leave these r-states[†], and hence the response property is not valid.

3.3. *Completeness of the Rule*

The above arguments implies that if the response property is valid, the algorithm will terminate properly without reporting "fail", i.e. Y is decreased to the empty set at each levels of the algorithm in the recursive computation of φ_i, F_i and Δ_i. We consider in turn each of the premises and show that it holds for the extracted constructs φ_i, F_i and Δ_i with $i \leq m$.

1 Premise of R_1 claims that every p-state s satisfies q or $r_j \wedge \varphi_j$, for some $j \in [1..n]$. Indeed, if s does not satisfy q, it belongs to the pend states (i.e., the initial Y, denoted hereafter by Y_0), and since all Y_0-states are divided and removed after the algorithm, s must be a part of the removed ones, it means that s belongs to some $r_j \wedge \varphi_j$.

2 Premise of R_2 requires that every immediate successor of s that satisfies $r_i \wedge \varphi_i$ must satisfy q or $r_j \wedge \varphi_j$ for some $j \in [1..n]$. As mentioned before, s belongs to Y_0, and its successor s_b must satisfy q or belong to Y_0. Similar to the situation in premise of R_1, we can get that s_b must belong to some $r_j \wedge \varphi_j$.

3 Premise of R_3 considers a state s_a that satisfies φ_i. Consider a successor s_b. It requires that s_b satisfies q, or φ_i and has the same value as $\Delta_i(s_a)$, or satisfies $r_j \wedge \varphi_j$ for some j and has $\Delta_j(s_b) \prec \Delta_i(s_a)$. According to the construction, every φ_i-state s has a rank $\Delta_i(s)$ and φ_i-state can be reached from a r_i-state by a finite φ_i-path π.
   — If s_b is a q-state, then it is acceptable.
   — If s_b is in Y, by the definition of $Y = \varphi_i + Y'$ (Y' is not reachable from states in φ_i) and the construction of φ_i, we know that s_b is in φ_i.
   — If s_b is in Y_0 - Y, such that s_b must have been removed from Y_0 in some earlier stage, satisfy $r_j \wedge \varphi_j$ with $\Delta_j(s_b) \prec \Delta_i(s_a)$ for some $j < i$.

4 Premise of R_4 requires that there is no transition from r_i-states to u_i-states can be taken in φ_i. By the definition of strengthened compassion , it satisfies this condition.

The above arguments proves the completeness, i.e., whenever a response property is valid, there exist auxiliary constructs for proving the property.

3.4. *Dealing with Systems with Infinite Number of States*

For dealing with infinite state systems, in addition to the above algorithm for constructing helpful assertions and ranks, we have to apply abstraction and concretization. The basic steps for proving the property is as follows

[†] Following the tradition of (Balaban et al., 2007), every state is assumed to have a loop to itself, and every state must be constrained by some fairness requirement in order to force the progress of computations of such a system model.

$$\left[\begin{array}{l} x : \textbf{natural}; \\ \textbf{init}\ x > 2; \\ l_0 : \textbf{while}: \ (x > 0) \\ \quad l_1 : \textbf{x:=2 or x:=x-1} \\ l_2 \end{array}\right]$$

Fig. 3. Non-Deterministic Choice

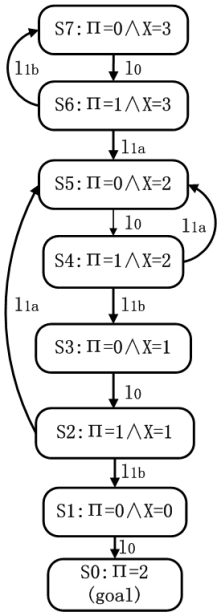

Fig. 4. Pend Graph of Non-Deterministic Choice

— Abstracting the program to a finite state one
— Extracting Auxiliary Constructs including the helpful assertions and ranking functions
— Concretizing the auxiliary constructs
— Applying the rule that uses the auxiliary constructs for proving the property

An example to demonstrate the above steps is shown in the next section.

4. An Example with Non-Deterministic Choice

Consider the program (which appeared also in (Main, 1993)) with a non-deterministic choice of the values of variable x in Figure 3.

At location l_1 there is a non-deterministic choice, and we may denote the first transition as l_{1a} and the second as l_{1b}. Let x' denote the next state variable of x.

The property we wish to establish is $at\_l_0 \Rightarrow \diamond at\_l_2$, and the strengthened compassion requirements are:

F_{c0}: $\langle at\_l_0, \{\neg at\_l_0\}\rangle$
F_{c1}: $\langle at\_l_1, \{at\_l_0 \wedge x \neq 2\}\rangle$
F_{c2}: $\langle at\_l_1, \{at\_l_0 \wedge x = 2\}\rangle$
F_{c3}: $\langle at\_l_1 \wedge x = 1, \{at\_l_0 \wedge x \neq 2\}\rangle$

The basic fairness requirement (constraining self-loop) of location 0 is F_{c0}. F_{c1} and F_{c2} implies that if it is at location 1 infinitely times, the action "x=x-1"must be taken from

$$\widetilde{x} = \begin{cases} 0 & x = 0 \\ 1 & x = 1 \\ 2 & x = 2 \\ 3 & x > 2 \end{cases}$$

location 1 infinitely times, so does action "x=2". F_{c3} implies that if it is at location 1 with x equals 1 infinitely times, the action "x=x-1" must be taken from location 1 with x equals 1 infinitely times.

For proving the property, the four steps described in the previous section are explained in the subsections as follows.

4.1. *Abstraction*

We apply the following abstraction α :

$$\alpha : \Pi = \pi, X = \widetilde{x}$$

where the assertion $\pi = i$ stands for at program location l_i. $\widetilde{x}$ is defined as follows.

The response property after applying the abstraction is now $\Pi = 0 \Rightarrow \diamond\Pi = 2$, and the strengthened compassion requirements are as follows:

$$\begin{array}{l} F_0 : \langle \Pi = 0, \{\Pi \neq 0\}\rangle \\ F_1 : \langle \Pi = 1, \{\Pi = 0 \wedge (X \neq 2)\}\rangle \\ F_2 : \langle \Pi = 1, \{\Pi = 0 \wedge (X = 2)\}\rangle \\ F_3 : \langle \Pi = 1 \wedge (X = 1), \{\Pi = 0 \wedge (X \neq 2)\}\rangle \end{array}$$

F_0, F_1, F_2 and F_3 are respectively the abstract version of F_{c0}, F_{c1}, F_{c2} and F_{c3}.

4.2. *Extracting Auxiliary Constructs*

The pend graph (containing all reachable states from p by $\neg q$-path and the transitions among them) of the finite state transition system after applying the abstraction is shown in Fig. 4. The abstracted helpful assertions Φ_i consisted by abstracted states S_i, values δ_i of ranking functions Δ_i on set of states $S \in \Phi_i$ and abstracted fairness requirements H_i which constraint Φ_i we get by algorithm *SC_Auxiliary_Constructs* are in Table. 1, and the process of the calculation is explained as follows.

1 We start with $rank\_SC(pend, [])$, and set Y= $pend$=$\{S_1, ..., S_7\}$, d=0, m=0.
2 Neither of F_1, F_2 and F_3 satisfies the assumption $\psi \wedge r$ at line 10. This means that these three strengthened compassion requirements may be fair to Y.
3 Then F_0 is chosen at line 8, and we get $\psi = \{S_1, S_2, S_4, S_6\}$.
4 At line 10, $(Y \wedge \Pi = 0) \neq \emptyset$, and we get $\varphi = \{S_1\}$ at line 13.
5 Set $m = 1$, $d = 1$, $\varphi_1 = \{S_1\}$, h_1= F_0, $\delta_1 = [1]$, Y'= $\{S_2, ..., S_7\}$ and $rem = 0$.
6 rem = 0 does not satisfy the condition at line 18, back to line 8 to choose another strengthened compassion requirement.
7 F_3 is chosen at line 8, and we get $\psi = Y = \{S_2, ..., S_7\}$.

Φ_i	S_i	$\delta_i = \Delta_i(S)$	H_i
Φ_7	$\Pi = 0 \wedge X = 3$	$[3,1]$	F_0
Φ_6	$\Pi = \{0,1\} \wedge X = 3$	$[3]$	F_2
Φ_5	$\Pi = 0 \wedge X = 2$	$[2,2,1]$	F_0
Φ_4	$\Pi = \{0,1\} \wedge X = 2$	$[2,2]$	F_1
Φ_3	$\Pi = 0 \wedge X = 1$	$[2,1]$	F_0
Φ_2	$\Pi = \{0,1\} \wedge X = 2$ $\vee \Pi = \{0,1\} \wedge X = 1$	$[2]$	F_3
Φ_1	$\Pi = 0 \wedge X = 0$	$[1]$	F_0

Table 1. *Abstract Assertions and Ranks of Non-Deterministic Choice*

8 At line 10, $(Y \wedge \Pi = 1 \wedge X = 1) \neq \emptyset$, and we get $\varphi = \{S_2, ..., S_5\}$ at line 13.

9 Set $m = 2$, $d = 2$, $\varphi_2 = \{S_2, ..., S_5\}$, $h_2 = F_3$, $\delta_2 = [2]$, $Y' = \{S_6, S_7\}$ and $rem = \{S_3, ..., S_5\}$.

10 Then call $rank\_SC(\{S_3, ..., S_5\}, [2])$ in which the following is done.

— Set $Y = \{S_3, ..., S_5\}$, d=0.
— Choose F_0 at line 8, we get $\psi = \{S_3, S_4\}$.
— At line 10, $(Y \wedge \Pi = 0) \neq \emptyset$, and we get $\varphi = \{S_3\}$ at line 13.
— Set m=3, d=1, $\varphi_3 = \{S_3\}$, $h_3 = F_0$, $\delta_3 = [2,1]$, $Y' = \{S_4, S_5\}$ and $rem = 0$.
— This continues until we have constructed φ_4, φ_5 and the respective ranks as shown in the table, and then Y in this recursion is empty.

11 After the recursive function call, we deal with $Y = \{S_6, S_7\}$, d=2 (equals the value of d before the recursion), m=5 (equals the value of m at the end of the recursion). Choose F_2 at line 8, and we get $\psi = \{S_6, S_7\}$.

12 At line 10, $(Y \wedge \Pi = 1) \neq \emptyset$, and we get $\varphi = \{S_6, S_7\}$ at line 13.

13 Set m=6, d=3, $\varphi_6 = \{S_6, S_7\}$, $h_6 = F_2$, $\delta_6 = [3]$, $Y' = \{\emptyset\}$ and $rem = \{S_7\}$.

14 Then call $rank\_SC(\{S_7\}, [3])$ in which the following is done.

— Set $Y = \{S_7\}$, d=0.
— Choose F_0 at line 8, we get $\psi = \{S_7\}$.
— At line 10, $(Y \wedge \Pi = 0) \neq 0$, and we get $\varphi = S_7$ at line 13.
— Set m=7, d=1, $\varphi_7 = \{S_7\}$, $h_7 = F_0$, $\delta_7 = [3,1]$, $Y' = \{\emptyset\}$ and $rem = 0$.
— In this recursion, Y is empty, back to the previous level.

15 Finally, Y is empty. The algorithm terminates successfully.

4.3. *Concretization*

The concrete helpful assertions φ_i consists by concrete states s_i and value of concrete states in ranking function $\Delta_i(s)$, and concrete fairness requirements h_i which constraint φ_i after concretization are shown in Table 2. The processes of concretization are as follows.

— *Concretizing Assertions*: This process changes the abstract variables back to the concrete ones by reversing the abstraction.
— *Concretizing Requirements*: This process changes the abstract requirements back to the corresponding concrete ones.

φ_i	s_i	$\Delta_i(s)$	h_i
φ_7	$at\_l_0 \wedge x > 2$	$[3, 1]$	F_{c0}
φ_6	$at\_l_{0,1} \wedge x > 2$	$[3]$	F_{c2}
φ_5	$at\_l_0 \wedge x = 2$	$[2, 2, 1]$	F_{c0}
φ_4	$at\_l_{0,1} \wedge x = 2$	$[2, 2]$	F_{c1}
φ_3	$at\_l_0 \wedge (x = 1)$	$[2, 1]$	F_{c0}
φ_2	$at\_l_{0,1} \wedge (0 < x \leq 2)$	$[2]$	F_{c3}
φ_1	$at\_l_0 \wedge (x = 0)$	$[1]$	F_{c0}

Table 2. *Concretized Assertions and Ranks of Non-Deterministic Choice*

4.4. *Application of the Rule*

The concrete helpful assertions φ_i and ranking functions Δ_i are the effective premises of the deductive rule SC_RESPONSE. They satisfy R_1-R_4, and therefore the property is proved to be true following the deductive rule. The readers are referred to the technical report (Long and Zhang, 2012) for the details on how to carry out such a proof.

5. Extending the Approach

The strengthened compassion requirements are very strong requirements. In some cases, when ranking abstraction is used, compassion requirements may be introduced, and we need a mixture of strengthened compassion requirements and compassion requirements for constraining the behavior in a system model. Therefore it needs deductive rules that can deal with such a mixture of fairness requirements.

5.1. *Fairness Discrete Systems*

A fairness discrete system (FDS) is quadruple $\mathcal{D} = \langle V, \Theta, \rho, \mathcal{F} \rangle$ where the components are as follows.

— V : A finite set of typed *system variables* - containing data and control variables. A set of states (interpretation) over V is denoted by Σ. For a state s and a system variable $v \in V$, we denote by $s[v]$ the value assigned to v by the state s.

— Θ : The *initial condition* - an assertion (state formula) characterizing the initial states.

— ρ : The *transition relation* - an assertion $\rho(V, V')$, relating the values V of the variables in state $s \in \Sigma$ to the values V' in a D-successor state $s' \in \Sigma$.
If φ is a formula representing a set of states, then φ' is the formula with each v replaced by v'.

— $\mathcal{F} = \{F_i \,|\, i = 1, ..., n\}$: The *fairness requirements* - a set of pairs of assertions where F_i is either an strengthened compassion requirement of the form $(r_i, \{u_i\})$ or a compassion requirement of the form (r_i, u_i). Let $\mathcal{C}$ denote the subset of $\mathcal{F}$ that contains the compassion requirements and $\mathcal{SC}$ denote the subset of $\mathcal{F}$ that contains the strengthened compassion requirements. The fairness requires that for each $i \in \{1, ..., n\}$, σ contains only finitely many r_i-states, or σ contains infinitely many r_i-states and if s_m is such a state (i.e., $s_m \models r_i$), then $\exists\ t = m + 1$ such that $s_t \models u_i$ (when $F_i = (r_i, \{u_i\}) \in \mathcal{SC}$ is a strengthened compassion requirement), or $\exists\ t > m$ such that $s_t \models u_i$ (when $F_i = (r_i, u_i) \in \mathcal{C}$ is a compassion requirement).

5.2. *Ranking Abstraction*

Ranking abstraction is a method of augmenting the concrete program by a non-constraining progress monitor, which measures the progress of program execution, relative to a given ranking function. In order to distinguish this kind of ranking functions from the ranking functions in the deductive rule, we call this kind of ranking functions ARFs (augmenting ranking functions) in the sequel. Once a program is augmented, a conventional state abstraction can be used. In such a way, the state abstraction can preserve the ability to monitor progress in the abstract system.

For a system FDS $\mathcal{D} = \langle V, \Theta, \rho, \mathcal{F} \rangle$ and a well-founded domain $(\mathcal{W}, \succ)$, let δ be an ARF over $\mathcal{W}$, let dec_δ be a fresh variable, the augmentation of $\mathcal{D}$ by δ is

$$D + \delta : \langle V \cup \{dec_\delta\}, \Theta, \rho \wedge \rho_\delta, F \cup \{\langle dec_\delta > 0, dec_\delta < 0 \rangle\} \rangle$$

where dec_δ is defined by

$$dec_\delta = \begin{cases} 1 & \delta \succ \delta' \\ 0 & \delta = \delta' \\ -1 & otherwise \end{cases}$$

A system may be augmented with a set of ARFs $\{\delta_1, ..., \delta_k\}$. Then predicate abstraction may be applied. In the predicate abstraction, it is not necessary to abstract variables of the form dec_δ since it ranges over the finite domain $\{-1, 0, 1\}$, and the abstraction preserves the compassion requirement $\langle dec_\delta > 0, dec_\delta < 0 \rangle$.

Let p, q be assertions.
Let $\mathcal{A} = (\mathcal{W}, \succ)$ be a well-founded domain.
Let $\{F_i \mid i \in \{1, ..., n\}\} \in \mathcal{C} \cup \mathcal{SC}$ be a set of fairness requirements.
Let $\{\varphi_i \mid i \in \{1, ..., n\}\}$ be a set of assertions.
Let $\{\Delta_i : \Sigma \to \mathcal{A} \mid i \in \{1, ..., n\}\}$ be a set of ranking functions.

R1	p	$\Rightarrow$	$q \vee \bigvee_{j=1}^{n}(r_j \wedge \varphi_j)$
$\forall i \leq n$:			
R2	$r_i \wedge \varphi_i \wedge \rho$	$\Rightarrow$	$q' \vee \bigvee_{j=1}^{n}(r'_j \wedge \varphi'_j)$
R3	$\varphi_i \wedge \rho$	$\Rightarrow$	$q' \vee (\varphi'_i \wedge \Delta_i = \Delta'_i) \vee \bigvee_{j=1}^{n}(r'_j \wedge \varphi'_j \wedge \Delta_i \succ \Delta'_j)$
R4	case $F_i = \langle r_i, u_i \rangle \in \mathcal{C}$:		
	φ_i	$\Rightarrow$	$\neg u_i$
	case $F_i = \langle r_i, \{u_i\} \rangle \in \mathcal{SC}$:		
	$\varphi_i \wedge r_i \wedge \rho \wedge \varphi'_i$	$\Rightarrow$	$\neg u'_i$
	p	$\Rightarrow$	$\diamond q$

Fig. 5. Deductive Rule: ESC_RESPONSE

5.3. *Deductive Rule: ESC_RESPONSE*

For dealing with both strengthened compassion and compassion, a deductive rule denoted ESC_RESPONSE is provided in Fig. 5. This rule extends SC_RESPONSE of Fig. 2 and is a combination of this and the rule RESPONSE of (Pnueli and Sa'ar, 2008) for dealing with compassion requirements. The correctness follows from the correctness of these two rules.

5.4. *Extended Algorithm*

Considering an FDS $\mathcal{D}$ with both compassion and strengthened compassion , we present an extended algorithm which extracts a deductive proof according to the rule ESC_RESPONSE of a response property $p \Rightarrow \diamond q$. The algorithm *E_Auxiliary_Constructs* is presented as Algorithm 2. The formulation of the algorithm involves a new operator $\mathbf{E}(p \; \mathcal{U} \; q)$ with the following meaning: the formula $\mathbf{E}(p \; \mathcal{U} \; q)$ captures the set of states that originate a path leading to any q-state, such that all the states in the path, except possibly the last, satisfy p. In this expression, the *until* temporal operator $\mathcal{U}$ is used.

Correctness The algorithm is a combination of the algorithm *SC_Auxilary_Constructs* for dealing with strengthened compassion constraints in $\mathcal{SC}$ and the algorithm in (Long and Zhang, 2010) for dealing with compassion constraints in $\mathcal{C}$. The correctness follows from that of the two algorithms, and can be proved similarly.

5.5. *Extended Example*

We consider the same example as that of section 4. With the use of ranking abstraction, a weaker fairness requirement is sufficient for proving the response property. We remove the strengthened compassion constraint $\langle at\_l_1, \{at\_l_0 \wedge x = 2\} \rangle$ from the original set

Algorithm 2 *E_Auxiliary_Constructs*

1: $m := 0$
2: $accessible_{\mathcal{D}} := E(trueS\Theta)$
3: $pend := accessible_{\mathcal{D}} \wedge E(\neg qS(p \wedge \neg q))$
4: $rank\_M(pend, [])$

where procedure $rank\_ESC$ is defined by:
procedure $rank\_ESC(subpart,prefix)$
d:**integer**
Y:**assertion**
5: Let $d := 0$
6: Let $Y := subpart$
7: FIX (Y)
8: **Forall** $(\langle r, u\rangle \in F)$ **do**
9: **if** $\langle r, u\rangle \in C$ **then**
10: Let $\psi = Y \wedge \neg(\mathbf{E}(Y\mathcal{U}(Y \wedge u)))$
11: **else if** $\langle r, u\rangle \in SC$ **then**
12: Let $\psi = Y \wedge \neg(Y \wedge r \wedge EX(Y \wedge u))$
13: **end if**
14: **if** $\psi \wedge r \neq \emptyset$ **then**
15: Let $m = m + 1$
16: Let $d = d + 1$
17: Let $\varphi_m := E(\psi S(\psi \wedge r))$
18: Let $h_m := \langle r, u\rangle_k$
19: Let $\delta_m := prefix * [d]$
20: Let $Y := Y \wedge \neg\varphi_m$
21: Let $rem := \varphi_m \wedge \neg r$
22: **if** $(rem \neq \emptyset)$ **then**
23: $rank\_ESC(rem, prefix * [d])$
24: **end if**
25: **end if**
26: end for
27: **if** $(Y \neq \emptyset)$ **then**
28: report "fail"
29: **end if**
30: **end-Fix**

of fairness requirements. The property we wish to establish is $at\_l_0 \Rightarrow \diamond at\_l_2$, and the fairness requirements are:

F_{ec0}: $\langle at\_l_0, \{\neg at\_l_0\}\rangle$
F_{ec1}: $\langle at\_l_1, \{at\_l_0 \wedge x \neq 2\}\rangle$
F_{ec3}: $\langle at\_l_1 \wedge x = 1, \{at\_l_0 \wedge x \neq 2\}\rangle$

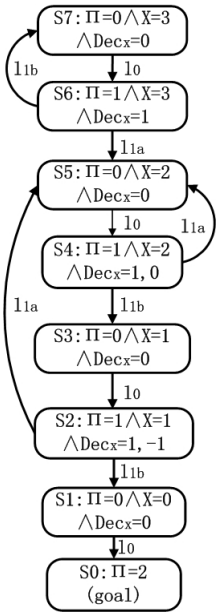

Fig. 6. Pend Graph of Non-Deterministic Choice with the Variable Dec_x

5.5.1. *Ranking Abstraction* The mapping for the abstraction is shown as follows.

$$\alpha : \Pi = \pi, X = \widetilde{x}, Dec_x = dec_x$$

$$\widetilde{x} = \begin{cases} 0 & x = 0 \\ 1 & x = 1 \\ 2 & x = 2 \\ 3 & x > 2 \end{cases} \qquad dec_x = \begin{cases} 1 & x \succ x' \\ 0 & x = x' \\ -1 & otherwise \end{cases}$$

The response property after applying the abstraction is now $\Pi = 0 \Rightarrow \diamond\Pi = 2$, and the fairness requirements are as follows:

$$\begin{array}{l} F_{e0} : \langle \Pi = 0, \{\Pi \neq 0\}\rangle \\ F_{e1} : \langle \Pi = 1, \{\Pi = 0 \wedge (X \neq 2)\}\rangle \\ F_{e3} : \langle \Pi = 1 \wedge (X = 1), \{\Pi = 0 \wedge (X \neq 2)\}\rangle \\ F_{Dec} : \langle Dec_x > 0, Dec_x < 0\rangle \end{array}$$

After removing a stronger constraint from concrete ones, instead, we add a weaker additional compassion requirement $F_{Dec} : \langle Dec_x > 0, Dec_x < 0\rangle$ (introduced by ranking abstraction) into the set of abstract requirements. F_{e0}, F_{e1} and F_{e3} are respectively the abstract version of F_{ec0}, F_{ec1} and F_{ec3}. The ranking abstraction introduces an extra fairness requirement, i.e., the compassion requirement $F_{Dec}\langle Dec_x > 0, Dec_x < 0\rangle$ which constraint that the value of x can not decrease infinitely times without increasing. The pend graph after the abstraction is shown in Fig. 6

5.5.2. *Extracting Auxiliary Constructs* By applying Algorithm 2 to the pend graph, we obtain the abstract auxiliary constructs in Table 3.

Φ_i	S_i	$\delta_i = \Delta_i(S)$	H_i
Φ_7	$\Pi = 0 \wedge X = 3 \wedge Dec_x = 0$	$[3,1]$	F_{e0}
Φ_6	$\Pi = 0 \wedge X = 3 \wedge Dec_x = 0$ $\vee \Pi = 1 \wedge X = 3 \wedge Dec_x = 1$	$[3]$	F_{Dec}
Φ_5	$\Pi = 0 \wedge X = 2 \wedge Dec_x = 0$	$[2,2,1]$	F_{e0}
Φ_4	$\Pi = 1 \wedge X = 2 \wedge Dec_x = \{0,1\}$ $\vee \Pi = 0 \wedge X = 2 \wedge Dec_x = 0$	$[2,2]$	F_{e1}
Φ_3	$\Pi = 0 \wedge X = 1 \wedge Dec_x = 0$	$[2,1]$	F_{e0}
Φ_2	$\Pi = 1 \wedge X = 2 \wedge Dec_x = \{0,1\}$ $\vee \Pi = 1 \wedge X = 1 \wedge Dec_x = \{-1,1\}$ $\vee \Pi = 0 \wedge X = 1 \wedge Dec_x = 0$ $\vee \Pi = 0 \wedge X = 2 \wedge Dec_x = 0$	$[2]$	F_{e3}
Φ_1	$\Pi = 0 \wedge X = 0 \wedge Dec_x = 0$	$[1]$	F_{e0}

Table 3. *Abstract Auxiliary Constructs of Non-Deterministic Choice with Dec_x*

5.5.3. *Concretization* Because Dec_x is an additional variable by ranking abstraction and must be remove after concretization, we should make sure that the requirement that includes Dec is reflected in the definition of ranks. The way to deal with this aspect is to append the value of variable x to corresponding ranking tuples (Balaban et al., 2007).

If a rank is associated with F_{Dec} requirement $\langle Dec_x > 0, Dec_x < 0\rangle$ then the rank Δ_i is to be modified as follows.

— If an assertion Φ_i is associated with an abstract fairness requirement $\langle Dec_x > 0, Dec_x < 0\rangle$, we get the concrete assertion $\varphi_i = \alpha^{-1}(\Phi_i \wedge Dec_x > 0)$ and change the h_i as the fairness requirement that $\varphi_i \models r_i$.

— If $\delta_i = [\beta]$ is obtained with an F_{Dec} requirement involving Dec_x, then we insert the value of variable x and 0 after β in δ_i. Then for $\delta_j = [\beta, \gamma]$ with the same prefix β, it will be modified to $[\beta, x_s, \gamma]$, i.e., $\Delta_j(s) = [\beta, x_s, \gamma]$ where x_s denotes the value of x at state s.

For instance, in Table 3, F_{Dec} is used to constrain φ_6 with rank $[\beta] = [4]$. Then $\delta_6 = [\beta] = [4]$ is modified to $\Delta_6(s) = [\beta, x_s, 0] = [4, x_s, 0]$. For $\delta_7 = [\beta, \gamma] = [4,1]$ with the same prefix "4", it is modified to $\Delta_7(s) = [4, x_s, 1]$.

After concretization, we obtain the concrete auxiliary constructs shown in Table. 4.

For states satisfying the same helpful assertions, the value of ranking functions depends

φ_i	s_i	$\Delta_i(s)$	h_i
φ_7	$at\_l_0 \wedge x > 2$	$[4, x_s, 1]$	F_{ec0}
φ_6	$at\_l_1 \wedge x > 2$	$[4, x_s, 0]$	F_{ec1}
φ_5	$at\_l_0 \wedge x = 2$	$[2, 2, 1]$	F_{ec0}
φ_4	$at\_l_{0,1} \wedge x = 2$	$[2, 2, 0]$	F_{ec1}
φ_3	$at\_l_0 \wedge (x = 1)$	$[2, 1]$	F_{ec0}
φ_2	$at\_l_{0,1} \wedge (0 < x \leq 2)$	$[2]$	F_{ec3}
φ_1	$at\_l_0 \wedge (x = 0)$	$[1]$	F_{ec0}

Table 4. *Concretized Auxiliary Constructs of Non-Deterministic Choice with Dec_x*

on the value of x in the states, and the value of x cannot decrease infinitely without increasing (constraint by F_{Dec} requirement).

5.5.4. *Application of the Rule* The concrete helpful assertions φ_i and ranking functions Δ_i are the effective premises of deductive proof rule ESC_RESPONSE. They satisfy R_1-R_4, and therefore the property is proved to be true following the deductive rule. The readers are referred to the technical report (Long and Zhang, 2012) for the details of such a proof.

6. Concluding Remarks

Strengthened compassion requirements have been studied in this paper. This kind of requirements has been compared with compassion requirements, and it shows the differences between the use of these two kinds of fairness requirements. A deductive rule SC_RESPONSE for proving response properties with such requirements has been presented. Proofs of the soundness and the relative completeness of the rule have also been provided, and the application of the rule has been illustrated. Furthermore, an extended rule ESC_RESPONSE has been provided for dealing with system models with a mixture of both compassion requirements and strengthened compassion requirements. The necessity and the use of such a rule have been demonstrated by an example.

References

Angluin, D., Aspnes, J., Fischer, M. J., and Jiang, H. (2005). Self-stabilizing population protocols. In *OPODIS*, pages 103–117.

Balaban, I., Pnueli, A., and Zuck, L. D. (2005). Ranking abstraction as companion to predicate abstraction. In *FORTE*, pages 1–12.

Balaban, I., Pnueli, A., and Zuck, L. D. (2007). Modular ranking abstraction. *Int. J. Found. Comput. Sci.*, 18(1):5–44.

Balaban, I., Pnueli, A., and Zuck, L. D. (2010). Proving the refuted: Symbolic model checkers as proof generators. In *Concurrency, Compositionality, and Correctness*, pages 221–236.

Ball, T., Majumdar, R., Millstein, T. D., and Rajamani, S. K. (2001). Automatic predicate abstraction of c programs. In *PLDI*, pages 203–213.

Fischer, M. J. and Jiang, H. (2006). Self-stabilizing leader election in networks of finite-state anonymous agents. In *OPODIS*, pages 395–409.

Graf, S. and Saïdi, H. (1997). Construction of abstract state graphs with pvs. In *CAV*, pages 72–83.

Kesten, Y., Pnueli, A., and on Raviv, L. (1998). Algorithmic verification of linear temporal logic specifications. In *ICALP*, pages 1–16.

Kesten, Y. and Pnueli, A. (2000). Verification by augmented finitary abstraction. *Inf. Comput.*, 163(1):203–243.

Kupferman, O. and Vardi, M. Y. (2005). From complementation to certification. *Theor. Comput. Sci.*, 345(1):83–100.

Lamport, L. (1977). Proving the correctness of multiprocess programs. *IEEE Trans. Software Eng.*, 3(2):125–143.

Lehmann, D. J., Pnueli, A., and Stavi, J. (1981). Impartiality, justice and fairness: The ethics of concurrent termination. In *ICALP*, pages 264–277.

Long, T. and Zhang, W. (2010). Auxiliary constructs for proving liveness in compassion discrete systems. In *ATVA*, pages 276–290.

Long, T. and Zhang, W. (2012). Proving liveness property under strengthened compassion requirements. In *TAMC*, pages 498–508.

Long, T. and Zhang, W. (2012). Proving Liveness Property under Strengthened Compassion Requirements. Technical Report, SKL–2012–01, Institute of Sofware, Chinese Academy of Sciences. Available at "http://lcs.ios.ac.cn/∼zwh/tr/".

Main, M. G. (1993). Complete proof rules for strong fairness and strong extreme fairness. *Theor. Comput. Sci.*, 111(1&2):125–143.

Manna, Z. and Pnueli, A. (1991). Completing the temporal picture. *Theor. Comput. Sci.*, 83(1):91–130.

Manna, Z. and Pnueli, A. (1994). Temporal verification diagrams. In *TACS*, pages 726–765.

Manna, Z. and Pnueli, A. (2010). Temporal verification of reactive systems: Response. In *Essays in Memory of Amir Pnueli*, pages 279–361.

Namjoshi, K. S. (2001). Certifying model checkers. In *CAV*, pages 2–13.

Namjoshi, K. S. (2003). Lifting temporal proofs through abstractions. In *VMCAI*, pages 174–188.

Peled, D., Pnueli, A., and Zuck, L. D. (2001). From falsification to verification. In *FSTTCS*, pages 292–304.

Peled, D. and Zuck, L. D. (2001). From model checking to a temporal proof. In *SPIN*, pages 1–14.

Pnueli, A. (1983). On the extremely fair treatment of probabilistic algorithms. In *STOC*, pages 278–290.

Pnueli, A. and Sa'ar, Y. (2008). All you need is compassion. In *VMCAI*, pages 233–247.

Sun, J., Liu, Y., Dong, J. S., and Pang, J. (2009). Pat: Towards flexible verification under fairness. In *CAV*, pages 709–714.