Extending Symmetry Reduction by Exploiting System Architecture

Richard Trefler[1]* and Thomas Wahl[2]**

[1] David R. Cheriton School of Computer Science, University of Waterloo, Canada
[2] Computer Systems Institute, ETH Zurich, Switzerland and
Oxford University Computing Laboratory, Oxford, United Kingdom

Abstract. Symmetry reduction is a technique to alleviate state explosion in model checking by replacing a model of replicated processes with a bisimilar quotient model. The size of the quotient depends strongly on the set of applicable symmetries, which in many practical cases allows only polynomial reduction. We introduce *architectural symmetry*, a concept that exploits architectural system features to compensate for a lack of symmetry in the system model. We show that the standard symmetry quotient of an architecturally symmetric and *well-architected* model preserves arbitrary Boolean combinations and nestings of reachability properties. This quotient can be exponentially smaller than the model, even in cases where traditional symmetry reduction is nearly ineffective. Our technique thus extends the benefits of symmetry reduction to systems that are in fact not symmetric. Finally, we generalize our results to all architecturally symmetric models, including those that are not well-architected. We illustrate our method through examples and experimental data.

1 Introduction

Symmetry is a feature of many multi-process systems that can be exploited in order to alleviate state explosion in model checking. A symmetry is a permutation of process indices that leaves the system model invariant. The idea of symmetry reduction is to replace the model by a smaller and bisimilar *quotient* that contains only one of the many states from the original model that are identical up to permutations. When the set of symmetries is large, the quotient model can be significantly smaller than the model under verification. In particular, under *full* symmetry the system model is invariant under all permutations of the process indices. This scenario is attractive for symmetry reduction as it allows an exponential reduction in model size.

Unfortunately, a system model may not be fully symmetric even when the system consists of replicated processes that execute the same parametrized program. Consider, for instance, protocols that assume a ring-like communication

* Supported by an Individual Discovery grant from NSERC, Canada.
** Supported by the EPSRC, United Kingdom, grant no. EP/G026254/1.

structure, in the style of the Dining-Philosophers resource allocation scheme. The fact that a process may only communicate with its neighbor to the "right" breaks full symmetry, as neighborhood is not preserved by arbitrarily rearranging processes. Similarly, consider the cache coherence problem in a modern multi-core hardware design. As the number of cores continues to grow, they will likely not be pairwise connected, but instead exhibit lean communication topologies, permitting only few process symmetries. As a result, symmetry reduction of a significant magnitude cannot be expected.

In this paper we address this lack of reduction by generalizing existing symmetry-based techniques for multi-process programs. More precisely, we show that full symmetry reduction is applicable to systems that are *not* fully symmetric, provided (i) the system model is *architecturally symmetric*, (ii) the system model is *well-architected*, and (iii) the property of interest is expressible in a rich subset of CTL called *Safety CTL*. Under these conditions, which we explain below, *architectural symmetry reduction* produces a quotient structure that is an *exact* abstraction and can be exponentially smaller than the traditional symmetry quotient.

Conditions (i) through (iii) form the core of our approach. An architectural symmetry of a structure M is a permutation of the process indices that leaves the *positive transitive closure* of M's transition relation R invariant, i.e. M's reachability relation R^+. In contrast, traditional symmetry reduction requires more strictly that R itself be invariant under permutations. A system model is well-architected if it provides, at all reachable system states, the possibility of return to an initial state. This property is common in reactive protocols that continuously respond to user requests, and in many non-terminating systems as a means to counter the effects of resource leaks. Finally, Safety CTL consists of all formulas built out of Boolean connectives and CTL's EF operator, including arbitrary nestings. We show that the *standard* symmetry quotient of a well-architected and architecturally symmetric system preserves—in both directions—Safety CTL properties. We formalize this preservation property in the notion of *safety bisimulation*, a relationship between structures that allows transitions in one system to be simulated by finite-length paths in the other and is therefore weaker than traditional bisimulation.

In summary, the contribution of this paper is to extend a well-known and popular technique, symmetry reduction, to a class of systems that the technique was previously considered inapplicable to. Given conditions (i) through (iii), any existing symmetry reduction technique can be applied to the model—no new reduction algorithm is required. This makes our technique combine effortlessly with existing tools. We finally extend our results to programs that are not well-architected: we show that architectural symmetry alone is enough to give rise to an exponentially smaller quotient that preserves—again in both directions—reachability properties.

2 Background

2.1 Kripke Structures

We use the standard nomenclature. Let AP be a finite set of *atomic propositions*. A *Kripke structure* is a tuple $M = (S, R, L, I)$, where S is a finite set of *states*, $R \subseteq S \times S$ is a set of *transitions* (state changes) of M, $L\colon S \to 2^{AP}$ is a *labeling function* that assigns to each state a set of atomic propositions considered *true* in that state, and finally $I \subseteq S$ is a set of designated initial states of M. A *path in M from s to t* is a sequence $(p^i)_{i=0}^k$ of states such that $k \geq 0$, $p^0 = s$, $p^k = t$ and $(p^i, p^{i+1}) \in R$ for all i with $0 \leq i < k$. The *length* of a path is the number of transitions in it. For instance, path $(p^i)_{i=0}^k$ has length k. A state t is *reachable* in M if there is a path in M from some initial state to t. We write R^+ for the positive transitive closure of R, i.e. R^+ is the smallest set such that

1. $R \subseteq R^+$, and
2. whenever $(u, v) \in R^+$ and $(v, w) \in R^+$, then also $(u, w) \in R^+$.

We have $(s, t) \in R^+$ exactly if there is a path in M from s to t of length at least 1.

Bisimulation. Let $M_1 = (S_1, R_1, L_1, I_1)$ and $M_2 = (S_2, R_2, L_2, I_2)$ be Kripke structures over AP. A relation $\approx \subseteq S_1 \times S_2$ is a *bisimulation* if $s_1 \approx s_2$ implies:

1. $L_1(s_1) = L_2(s_2)$,
2. for every $t_1 \in S_1$ such that $(s_1, t_1) \in R_1$, there exists $t_2 \in S_2$ such that $t_1 \approx t_2$ and $(s_2, t_2) \in R_2$, and
3. for every $t_2 \in S_2$ such that $(s_2, t_2) \in R_2$, there exists $t_1 \in S_1$ such that $t_1 \approx t_2$ and $(s_1, t_1) \in R_1$.

If $\approx$ is a bisimulation, and for each $s_1 \in I_1$ there exists $s_2 \in I_2$ such that $s_1 \approx s_2$, and for each $s_2 \in I_2$ there exists $s_1 \in I_1$ such that $s_1 \approx s_2$, then M_1 and M_2 are *bisimilar*. Bisimilarity implies that the structures satisfy the same properties of the temporal logic *CTL*. CTL is the smallest set of formulas that comprises *false*, *true*, the atomic propositions (AP), and is closed under Boolean connectives and the *temporal modalities* EX, AX, EF, EG, EU, etc.

Canonical Quotients. Many existential abstractions are based on the formation of a *canonical* quotient of the given Kripke structure, as follows. Let $M = (S, R, L, I)$ and $\equiv$ be an equivalence relation on S such that $s \equiv t$ implies $L(s) = L(t)$, with equivalence classes written as $[s]$. The canonical quotient of M is given by the structure $M' = (S', R', L', I')$ such that

$$\begin{aligned} S' &= \{\, [s] : s \in S \,\}, \\ R' &= \{\, ([s], [t]) \in S' \times S' : \ \exists s_0 \in [s],\, t_0 \in [t] : \ (s_0, t_0) \in R \,\}, \\ L'([s]) &= L(s), \text{ and} \\ I' &= \{[s] : s \in I\}\,. \end{aligned}$$

The requirement that $s \equiv t$ imply $L(s) = L(t)$ ensures that L' is well-defined. As an example, let $\equiv_L$ be the *labeling equivalence* with respect to L, i.e. the relation on S defined by $s \equiv_L t$ iff $L(s) = L(t)$. Relation $\equiv_L$ is the *coarsest* equivalence relation that allows L' to be well-defined.

2.2 Symmetry in Multi-Process Systems

The term "process" is used in this paper generically for a component of a concurrent system. A *state* $(\vec{g}, l_1, \ldots, l_n)$ in such a system consists of the values $\vec{g}$ of all global variables (not associated with any process) and the *local state* l_i of each process $i \in \{1, \ldots, n\}$ (values of all local variables of process i).

Symmetries of a Kripke model M are defined with respect to permutations (bijections) $\pi\colon S \to S$ on the state space S; we describe such permutations in more detail in the next paragraph. We extend π to a mapping $\pi\colon R \to R$ on the transition level by defining $\pi((s,t)) = (\pi(s), \pi(t))$.

Definition 1 *A permutation π on S is said to be a **symmetry** of Kripke structure $M = (S, R, L, I)$ if*

1. *R is invariant under π: $\pi(R) = R$, and*
2. *L is invariant under π: $L(s) = L(\pi(s))$ for any $s \in S$, and*
3. *I is invariant under π: $\pi(I) = I$.*

*The symmetries of M form a group under function composition. Model M is said to be **symmetric** if its symmetry group G is non-trivial; we speak of symmetry **with respect to** G.*

For an n-process system, a symmetry π is derived from a permutation on $\{1, \ldots, n\}$ and acts on a state $s = (\vec{g}, l_1, \ldots, l_n)$ as $\pi(s) = (\vec{g}^{\,\pi}, l_{\pi(1)}, \ldots, l_{\pi(n)})$. That is, the local states of the processes are permuted by permuting their positions in the state vector. Further, π acts on $\vec{g}$ by acting component-wise on each global variable g. The action of π on g depends on the nature of g; we refer the reader to [8] for details.

Exploiting symmetry. Given a group G of symmetries, the relation $s \equiv_o t$ iff $\exists \pi \in G : \pi(s) = t$ defines an equivalence between states and is known as the *orbit relation*; the equivalence classes it entails are called *orbits* [4], written $[s]$ for $s \in S$. Observe that $s \equiv_o t$ implies $L(s) = L(t)$, since L is invariant under permutations in G (Definition 1). Let therefore $\overline{M}$ be the canonical quotient of M with respect to $\equiv_o$. Quotient $\overline{M}$ turns out to be bisimilar to M [4]. As a result, for two states $s \in S$, $\overline{s} \in \overline{S}$ with $s \in \overline{s}$ and any CTL formula f over AP whose atomic propositions are invariant under permutations in G,

$$M, s \models f \quad \text{iff} \quad \overline{M}, \overline{s} \models f. \tag{1}$$

Depending on the size of G, $\overline{M}$ can be up to exponentially smaller than M. For example, for full symmetry in n-process systems, all $n!$ many permutations of a global state with pairwise distinct local states are orbit-equivalent and can be collapsed into a single abstract state.

3 Safety Bisimulation

The goal of this paper is to dramatically reduce the verification complexity for certain systems with only little symmetry. The quotients that we obtain can

therefore not be expected to be bisimilar to the original system model. Instead, we will use the following weaker notion.

Definition 2 *Let $M_1 = (S_1, R_1, L_1, I_1)$ and $M_2 = (S_2, R_2, L_2, I_2)$ be Kripke structures over AP. Relation $\approx_r \subseteq S_1 \times S_2$ is a* ***safety bisimulation*** *if $s_1 \approx_r s_2$ implies:*

1. *$L_1(s_1) = L_2(s_2)$,*
2. *for every $t_1 \in S_1$ such that $(s_1, t_1) \in R_1$, there exists $t_2 \in S_2$ such that $t_1 \approx_r t_2$ and $(s_2, t_2) \in R_2^+$, and*
3. *for every $t_2 \in S_2$ such that $(s_2, t_2) \in R_2$, there exists $t_1 \in S_1$ such that $t_1 \approx_r t_2$ and $(s_1, t_1) \in R_1^+$.*

If $\approx_r$ is a safety bisimulation, and for each $s_1 \in I_1$ there exists $s_2 \in I_2$ such that $s_1 \approx_r s_2$, and for each $s_2 \in I_2$ there exists $s_1 \in I_1$ such that $s_1 \approx_r s_2$, then M_1 and M_2 are ***safety-bisimilar****.*

Safety bisimilarity is identical to bisimilarity except for the occurrences of R_2^+ and R_1^+ in conditions 2 and 3. Figure 1 shows pairs of safety-bisimilar structures that are not bisimilar. The safety bisimulation relates states with identical labels.

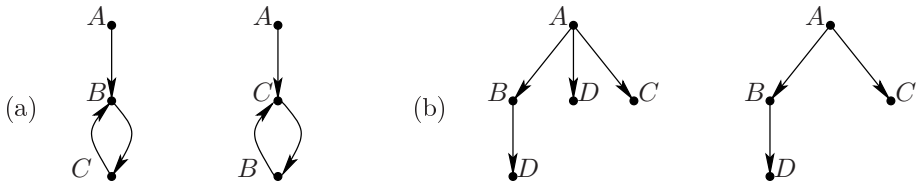

Fig. 1. Examples of safety-bisimilar structures

As with bisimilarity and CTL, there is a temporal logic that cannot distinguish safety-bisimilar structures.

Definition 3 ***Safety CTL****, denoted by EF-CTL, is the smallest set of formulas satisfying the following conditions:*

base formulas: *The Boolean constants false and true are EF-CTL formulas. For $P \in AP$, P is an EF-CTL formula.*
closure under Boolean connectives: *If f is an EF-CTL formula, so is $\neg f$. If g and h are EF-CTL formulas, so are $g \wedge h$, $g \vee h$, etc.*
closure under EF: *If f is an EF-CTL formula, so is $\mathsf{EF}\, f$.*

By definition, EF-CTL is a strict subset of CTL: neither *next-time* nor *until* operators can be expressed in (or generally translated into) EF-CTL. On the other hand, as common in CTL we use $\mathsf{AG}\, f$ as an abbreviation for $\neg\, \mathsf{EF}\, \neg f$. Thus, a CTL formula belongs to EF-CTL if any modality occurring in it is

EF or AG. For instance, consider a system with two processes i and j and the property that it always be possible to reach a state in which the processes are synchronized. This property, which is neither of the classical safety nor liveness type, is expressed in EF-CTL as $\mathsf{AG}\,\mathsf{EF}\,\mathit{synch}(i,j)$.

Since EF-CTL is a sub-logic of CTL, we can define its semantics by resorting to CTL. We write $M, s \models f$ to mean that the EF-CTL formula f evaluates to *true* over structure M and state s, which in turn is to mean that with CTL semantics, $M, s \models f$.

We now establish the relationship between safety bisimilarity and EF-CTL:

Theorem 4 *Let $M_1 = (S_1, R_1, L_1, I_1)$ and $M_2 = (S_2, R_2, L_2, I_2)$ be Kripke structures over AP and $\approx_r$ a safety bisimulation between them. Let further s_1, s_2 be states with $s_1 \approx_r s_2$ and f be an EF-CTL formula. Then $M_1, s_1 \models f$ exactly if $M_2, s_2 \models f$.*

Proof: We show the $\Rightarrow$ direction; the inverse direction follows since the inverse relation $\approx_r^{-1} \subseteq S_2 \times S_1$ is a safety bisimulation as well. The proof is by induction on the structure of f.

(base formulas)
The interpretations of *false* and *true* are independent of M_1, s_1, M_2, s_2. Let $f \in AP$. From $s_1 \approx_r s_2$, it follows that $L_1(s_1) = L_2(s_2)$ and therefore $M_1, s_1 \models f$ implies $f \in L_1(s_1)$, hence $f \in L_2(s_2)$ and thus $M_2, s_2 \models f$.

(closure under Boolean connectives)
The result follows immediately from the induction hypothesis and the semantics of $\neg$, $\wedge$, $\vee$.

(closure under EF)
Suppose $M_1, s_1 \models \mathsf{EF}\,g$. This means that there is a path $p := (p^i)_{i=0}^k$ in M_1 with $p^0 = s_1$ and $M_1, p^k \models g$. We now claim that there exists a state $q \in S_2$ that is reachable from s_2 and satisfies $p^k \approx_r q$. Given this claim and $M_1, p^k \models g$, we apply the induction hypothesis to conclude that $M_2, q \models g$. Since q is reachable from s_2, this proves $M_2, s_2 \models \mathsf{EF}\,g$.
To show the claim, we proceed by induction on k. If $k = 0$, then $p^k = p^0 = s_1$. Choosing $q := s_2$ satisfies all requirements.
Assume now p has the form $(p^i)_{i=0}^{k+1}$, and consider the prefix $(p^i)_{i=0}^k$ of p. By the induction hypothesis (of the claim), there exists a state $q' \in S_2$ that is reachable from s_2 and satisfies $p^k \approx_r q'$. Further, $(p^k, p^{k+1}) \in R_1$. Since $\approx_r$ is a safety bisimulation between M_1 and M_2, there is a state $q \in S_2$ with $(q', q) \in R_2^+$ and $p^{k+1} \approx_r q$. In particular, q is reachable from q' and thus reachable from s_2 and satisfies all requirements of the claim. □

Figure 1 demonstrates that the addition of CTL's *next-time* (X) or *until* (U) operators is enough to distinguish safety-bisimilar structures. The CTL formula $\mathsf{EX}\,B$ is true of the first structure in (a), but not of the second. Likewise, the CTL formula $\mathsf{E}((\mathsf{EF}\,C)\,\mathsf{U}\,D)$ is true of the first structure in (b), but not of the second. In contrast, the two structures in (a) satisfy the same EF-CTL formulas, as do the two structures in (b).

4 Architectural Symmetry

We are now ready to define architectural symmetry and show that, under certain conditions, it permits a safety-bisimilar quotient. Looking back at Definition 1 (symmetry), the crucial property of a symmetric model is its invariance under permutations. Since safety bisimilarity is weaker than bisimilarity, we can afford a weaker invariance notion, thus capturing a larger class of systems.

Definition 5 *A permutation π on S is said to be an **architectural symmetry** of the Kripke structure $M = (S, R, L, I)$ if*

1. *R^+ is invariant under π: $\pi(R^+) = R^+$,*
2. *L is invariant under π: for any $s \in S$, $L(s) = L(\pi(s))$, and*
3. *I is invariant under π: $\pi(I) = I$.*

*The architectural symmetries of M form a group under function composition. Structure M is said to be **architecturally symmetric** if its architectural symmetry group G is non-trivial; we speak of architectural symmetry **with respect to** G.*

Note the difference to Definition 1: in item 1, instead of requiring R to be permutation invariant, we require R^+ to be. Consider a finite path p between two states s and t. Under symmetry, the permuted sequence $\pi(p)$ is a valid path as well, connecting $\pi(s)$ and $\pi(t)$. Under architectural symmetry, all we can say is that $\pi(s)$ and $\pi(t)$ are connected in M as well, since $(\pi(s), \pi(t)) \in \pi(R^+) = R^+$.

Comparing the two types of symmetry, we confirm that architectural symmetry is weaker than symmetry:

Lemma 6 *If M is symmetric with respect to a group G, then M is architecturally symmetric with respect to G.*

Proof: We have to show that each symmetry is an architectural symmetry. Let $\pi \in G$ be a permutation on S. Requirements 2 and 3 are identical in definitions 1 and 5. Regarding requirement 1, suppose $\pi(R) = R$, we show $\pi(R^+) \subseteq R^+$ (the other inclusion follows with a symmetric argument; note that R^+ is a finite set).

To this end, consider $(s_1, t_1) \in \pi(R^+)$, i.e. $(s_1, t_1) = \pi((s_2, t_2))$ for some pair $(s_2, t_2) \in R^+$. Let p_2 be a path in M that connects s_2 to t_2. Each transition of p_2 belongs to R. Therefore, each transition of $p_1 := \pi(p_2)$ belongs to $\pi(R) = R$, so p_1 is a valid path. Since p_1 connects $\pi(s_2) = s_1$ to $\pi(t_2) = t_1$, it follows that $(s_1, t_1) \in R^+$. □

Example. In the following we demonstrate that Lemma 6 can in general not be strengthened to an equivalence. Consider a token ring model where the shared token regulates access to some resource. Such rings occur in hardware models and in communication protocols. Figure 2 shows the local transition diagram of process i. The process may be in one of the local states N, N^+, T, T^+, C; there are no global variables. Intuitively, N, T and C indicate that the process is "not trying" to access the resource, "trying" to do so, or is "currently"

accessing it. The superscript $^+$ indicates ownership of the token. The process can move freely between local states N and T, and also between N^+ and T^+. To acquire the token, it must currently be possessed by the left neighbor, process $i-1$ ($i-1$ and $i+1$ are defined cyclically within the index range $\{1,\ldots,n\}$), and that neighbor must be willing to release the token. This is indicated by the simultaneous transition $N_{i-1}^+ \to N_{i-1}$ in Figure 2. Analogously, to release the token it must be received by process $i+1$, which must be ready to do so (indicated by N_{i+1} or T_{i+1}). Let finally

$$I \quad := \quad \{(s_1,\ldots,s_n) : \ \exists i : s_i = N^+ \ \wedge \ \forall j : j \neq i : s_j = N\}$$

be the set of initial states: every process is non-trying, and any one of them owns the token.

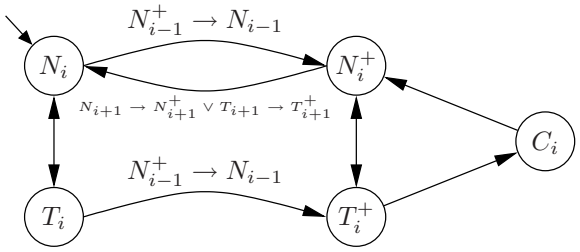

Fig. 2. Token ring example for resource allocation lacking full symmetry

It is easy to prove that the Kripke structure M induced by the parallel composition of n processes running the program in Figure 2 enjoys rotational symmetry: permutations of the cycle form $(1\ 2\ \ldots\ n)$ leave the structure invariant. M is not, however, fully symmetric: consider $n=3$ and the transition

$$\tau \quad := \quad (N_1^+,\ T_2,\ T_3) \quad \to \quad (N_1,\ T_2^+,\ T_3)\,.$$

Applying the transposition $(1\ 2)$ to τ results in the two states $(T_1,\ N_2^+,\ T_3)$ and $(T_1^+,\ N_2,\ T_3)$. The transition from T_1 to T_1^+ is not allowed by Figure 2 in the context of state $(T_1,\ N_2^+,\ T_3)$. In fact, consider any permutation π such that $\pi(i-1)+1 \neq \pi(i)$. Then π is not a symmetry of M, by the same argument. As a consequence, for a symmetry π the condition $\pi(i-1)+1 = \pi(i)$ is necessary for all i, and it is also sufficient. Thus, the rotation group is the largest symmetry group of M, and one cannot expect more than linear savings due to standard symmetry reduction for this structure.

On the other hand, applying Definition 5, we see that M is *fully* architecturally symmetric. We first show that $\pi(R) \subseteq R^+$. The only interesting cases are the transitions where process i acquires the token from its left neighbor $i-1$. After permuting such a transition, the process $\pi(i-1)$ releasing the token is generally someone other than the left neighbor. The resulting invalid transition

can be simulated by a path that passes the token from $\pi(i-1)$ successively to that process' right neighbors until it eventually reaches process $\pi(i)$ (some temporary moves from T to N may be required to enable the passing of the token). To show that $\pi(R^+) = R^+$, one applies this idea to each transition of a permuted path and connects the resulting paths to a (long) final path. Also, I is invariant under arbitrary permutations, and L can be defined to be so. In conclusion, M features an exponential-size architectural symmetry group, but only a small polynomial-size standard symmetry group. (End of example.)

Before we demonstrate the benefits of reducing architecturally symmetric systems in the next section, we show the following property.

Lemma 7 *Let M be architecturally symmetric with respect to G and Reached be the set of reachable states of M. Then, for all $\pi \in G$, $\pi(Reached) = Reached$.*

Proof: We show $\pi(Reached) \subseteq Reached$ (the other inclusion follows with a symmetric argument; note that *Reached* is a finite set). Assume $\pi(R^+) = R^+$, and consider $t \in \pi(Reached)$, i.e. $t = \pi(r)$ for some $r \in Reached$. Then there is some initial state $s \in I$ such that $(s, r) \in R^+$, i.e. $\pi(s, r) = (\pi(s), \pi(r)) = (s', t) \in \pi(R^+) = R^+$, where $s' := \pi(s)$. Since, by Definition 5, I is invariant under π, it follows that $s' \in I$. Thus, t is reachable in M (namely, from s'), i.e. $t \in Reached$. □

5 Well-Architected Systems

Architectural symmetry alone is not yet enough to permit a safety-bisimilar quotient. We therefore now consider models satisfying the following condition:

Definition 8 *A system model $M = (S, R, L, I)$ is **well-architected** if*

1. *M's initial states are all reachable from each other, and*
2. *for every reachable state s, there is an initial state that is reachable from s.*

The possibility of returning to the initial state at any time is common in reactive systems to prevent resource leaks in long-running executions, for instance through *micro-reboots* [2]. Communication protocols in the IP and telephony communities regulate the coexistence of interacting *features*. Each feature is a finite-state terminating process; overall behavior is described by continuously selecting an appropriate feature based on current input, executing the feature to completion, issuing appropriate output and then returning to some (often the unique) initial state.

As a concrete example, the model of the resource allocation scheme shown in Figure 2 is well-architected: first, the initial states are reachable from each other, since the token can be passed around until it reaches which ever process requested to have it. Second, *every* initial state can be reached from any reachable state by letting each process individually return to its initial local state (N or N^+); the token may again have to be passed around to whoever held it initially.

From the definition of well-architectedness, we conclude:

Observation 9 *Let $M = (S, R, L, I)$ be well-architected and u and v be reachable states. Then u and v are reachable from each other.*

Proof: Since M is well-architected, there is a path from u to some $s \in I$. Since v is reachable, there is a path from some $t \in I$ to v. Again since M is well-architected, s and t are mutually reachable. Putting it all together, there is a path from u to v. The reachability of u from v follows symmetrically. □

Architectural Symmetry Quotients of Well-Architected Systems

We now present the main result of this paper: From a well-architected and architecturally symmetric model M, one can derive a safety-bisimilar quotient structure M'. Quotient M' is obtained as the canonical quotient (see Section 2.1) of M with respect to the orbit relation $\equiv_o$ on S. In other words, EF-CTL formulas can be verified reliably over the standard symmetry quotient, although the underlying model is *not* symmetric.

Theorem 10 *Let M be well-architected and architecturally symmetric with respect to G. Let further $\equiv_o$ be the orbit relation on S, i.e. $s \equiv_o t$ iff $\exists \pi \in G : \pi(s) = t$. Let finally M' be the canonical quotient of structure M with respect to $\equiv_o$. Then M' is safety-bisimilar to M.*

Proof: We first remark that the labeling function of the quotient is well-defined: Let $s \equiv_o t$. Then there exists $\pi \in G$ such that $\pi(s) = t$. Since M is architecturally symmetric, L is invariant under π, which implies $L(s) = L(t)$.

We now define a suitable relation $\equiv_r$ between S and S', namely:

$$s_1 \equiv_r [s_2] \quad \text{iff} \quad s_1 \equiv_o s_2 \,.$$

In particular, $s \equiv_r [s]$ for any $s \in S$. We claim that $\equiv_r$ is a safety bisimulation. The theorem then follows, since the initial states of M and M' are appropriately related: for any $s \in I$, it is $s \equiv_r [s] \in I'$. Further, for any $[s] \in I'$, it is $[s] \equiv_r s \in I$. To show the claim, let $s_1 \equiv_r [s_2]$, hence $s_1 \equiv_o s_2$ and thus $[s_1] = [s_2]$. We prove the three conditions of Definition 2. We restrict our attention to the reachable part of M, which is commonly achieved by exploring M and building the quotient on the fly. That is, s_1 is an actually reached and, therefore, reachable state of M.

1. By the remark above about $\equiv_o$, we obtain $L(s_1) = L(s_2)$, and by the definition of M', $L(s_2) = L'([s_2])$. Thus $L(s_1) = L'([s_2])$.
2. Let t_1 be such that $(s_1, t_1) \in R$. We choose $t_2 := t_1$ and consider $[t_2]$: It is $t_1 \equiv_r [t_1] = [t_2]$. Further, from $(s_1, t_1) \in R$ we conclude $([s_1], [t_1]) = ([s_2], [t_2]) \in R' \subseteq R'^{+}$.
3. Let $[t_2]$ be such that $([s_2], [t_2]) \in R'$. By definition of R', there exist $s \in [s_2]$, $t \in [t_2]$ such that $(s, t) \in R$. We conclude $t \equiv_r [t_2]$. Further, from $s_1 \equiv_o s_2$ and $s \equiv_o s_2$, we conclude $s_1 \equiv_o s$. By Lemma 7 and the reachability of s_1, it follows that s is also reachable in M. Therefore t is reachable in M. Since

s_1 is also reachable in M, by Observation 9 there is a path from s_1 to t, which implies $(s_1, t) \in R^+$. We now choose $t_1 := t$ to obtain $t_1 = t \equiv_r [t_2]$ and $(s_1, t_1) = (s_1, t) \in R^+$.

Note that well-architectedness was used only in the form of Observation 9. □

We summarize this section in the following statement:

Corollary 11 *M and M' as in Theorem 10 satisfy the same EF-CTL properties.*

We emphasize again that, assuming G is the full symmetry group, M' is exponentially smaller than M, although standard symmetry may allow only an insignificant reduction. Note, however, that in order to apply full architectural symmetry reduction, the EF-CTL properties of interest must have fully symmetric atomic propositions.

Consider again the example in Figure 2, which allows only polynomial symmetry reduction: Since it is both well-architected and architecturally symmetric with respect to the full symmetry group, we can apply full symmetry reduction to it when verifying EF-CTL formulas, giving rise to an exponentially smaller quotient. In Section 7 we underpin this result with quantitative data.

6 Generalization: Non-Well-Architected Systems

We briefly demonstrate that well-architectedness is not even required if one restricts the set of eligible formulas further, namely to reachability properties:

Theorem 12 *Let $M = (S, R, L, I)$ be architecturally symmetric with respect to group G, and let M' be the canonical quotient of M with respect to the orbit relation. For any state $s \in S$ and an atomic proposition q,*

$$M, s \models \mathsf{EF}\, q \quad \textit{iff} \quad M', [s] \models \mathsf{EF}\, q\,.$$

Analogously, the theorem can be stated as $M, s \models \mathsf{AG}\, q$ iff $M', [s] \models \mathsf{AG}\, q$. In other words, for architecturally symmetric systems, safety properties such as the unreachability of an error state can be equivalently formulated over the quotient M'. To prove the theorem, we first show a stronger *path correspondence* result. It addresses the "disconnect problem" of existential abstractions, namely that in general a path in the abstract system may not be liftable to one in the concrete system. It turns out that under architectural symmetry, it is.

Lemma 13 *Let M and M' be as above and $(p'^i)_{i=0}^k$ be a path in M'. Then, for any $s \in p'^0$, there is a path in M from s to some element $t \in p'^k$.*

Proof: By induction on k. If $k = 0$, choose $t := s$ to get a path in M of length 0. Now consider path $(p'^i)_{i=0}^{k+1}$, and let $s \in p'^0$. By the induction hypothesis, there is a path p in M from s to some state $t^k \in p'^k$. Further, $(p'^k, p'^{k+1}) \in R'$ implies that there is a transition $(x, y) \in R$ such that $x \in p'^k$, $y \in p'^{k+1}$. Then

$x \equiv_o t^k$, so let $\pi \in G$ be a permutation such that $\pi(x) = t^k$. By architectural symmetry, $(x, y) \in R \subseteq R^+ = \pi(R^+)$, thus $(\pi(x), \pi(y)) = (t^k, \pi(y)) \in R^+$. Concatenating path p and the path from t^k to $\pi(y)$ results in a path in M from s to $t^{k+1} := \pi(y) \in p'^{k+1}$. □

Proof [Theorem 12]:

"⇒": Any path in M from s to t satisfying q can be mapped to a path in M' from $[s]$ to $[t]$. By the definition of L', $q \in L(t) = L'([t])$.

"⇐": Suppose $M', [s] \models \mathsf{EF}\, q$, i.e. there is a path p' in M' with $p'^0 = [s]$ and $q \in L'(p'^k)$ for some k. By Lemma 13, since $s \in [s] = p'^0$, there is a path in M from s to some element $t \in p'^k$. Thus, $q \in L'(p'^k) = L(t)$ by requirement 2 of Definition 5, proving $M, s \models \mathsf{EF}\, q$. □

7 Experiments and Further Examples

In this section we present some quantitative data to support our proposed technique. We consider the token ring example from Section 4 and show the difference between model checking this system by exploiting standard symmetry, and by exploiting architectural symmetry. We have already established that the Kripke structure induced by the system is rotationally symmetric, and that it is also both well-architected and architecturally symmetric with respect to the full symmetry group.

We conducted experiments using the SVISS symbolic verifier [17], an experimental platform for symmetric systems. SVISS is based on the CUDD BDD library [15] and supports various symmetry groups, in particular the rotational and the full group, which are relevant for our example. We ran the example on a 2GB main memory dual-core 2.2GHz system. The property we verified is mutually exclusive occupancy of the C local state. This property is satisfied on this system, so that SVISS generates the full reachable state space (up to symmetry reduction).

		Rotational Symmetry		Architectural Symmetry	
Numb. of processes	BDD nodes Trans. Rel.	Numb. of BDD nodes	Time	Numb. of BDD nodes	Time
40	1,647	46,103	0:07m	5,612	0:02m
50	2,067	70,448	0:23m	7,052	0:06m
60	2,487	99,893	0:53m	8,492	0:11m
70	2,907	134,438	1:36m	9,932	0:21m
80	3,327	174,083	2:48m	11,372	0:35m
90	3,747	218,828	4:33m	12,812	0:58m
100	4,167	268,673	6:59m	14,252	1:24m

Table 1. Space and time requirements for the token ring example

The table shows, for a growing number of processes executing the protocol, the size of the BDD for the transition relation, the maximum number of live BDD nodes ("Numb. of BDD nodes") during the verification run, and the running time. This example, although small, does impart the difference an exponential reduction makes over a polynomial one, namely the potential to scale up to large examples, especially regarding memory, the classical bottleneck of BDDs.

Architectural symmetry and multi-core memory consistency. With the advent of multi-core hardware designs, pairwise connected communication topologies will be too costly to support. Instead, hardware and software communication topologies based on rings, tori, trees, hypercubes, and specially designed patterns will likely abound [1]. No matter what the exact topology, it will be necessary to ensure some form of data consistency among the various cores accessing a shared memory segment. That is, for a particular memory location accessed by several cores, and possibly several internal core-level caches, the values stored in those processor locations should be consistent.

Violation of multi-core memory consistency can be stated using a formula of the form $\mathsf{EF}(\exists i, j : v(p_i) \neq v(p_j))$, expressing the reachability of a state where two processors p_i, p_j have different values for a single memory location v. This formula has fully symmetric atomic propositions, so that architectural symmetry reduction techniques can be applied in a straightforward manner. As a consequence, the property can be verified over architecturally symmetric systems (whether well-architected or not), enjoying the same reduction as fully symmetric ones, namely with an exponentially smaller quotient.

FlexRay, Time-triggered architectures. In the automotive electronics industry, the *FlexRay* consortium has been formed by major car manufacturers to design a communication protocol for the control logic in vehicles [9]. Bus and star networks are supported, as well as any hybrid topology resulting from a combination of bus and stars. Many dozens of nodes can be connected in a FlexRay network, making full interconnection too expensive. Similar structures with little conventional symmetry are supported by the *time-triggered protocol*, where the network is a broadcast bus, often equipped with dual channels for fault tolerance [10].

8 Conclusion and Outlook

We have described a new notion of *architectural symmetry*, which extends attractive benefits of symmetry reduction to many systems with little symmetry. The result is a potential for an exponentially more effective reduction in model size. The price we pay is an architectural requirement of *well-architectedness* and a specification language with less expressive power than CTL, namely EF-CTL. We have given examples of multi-process systems that can be fully symmetry-reduced, although the model under verification is only rotationally symmetric. We have finally shown that the requirement of well-architectedness can be traded in for a restriction to reachability properties.

Relation to previous work. Symmetry reduction for model checking was introduced in [4, 6], and in [11] using *scalarsets* for fully symmetric systems. These works demonstrate the potential of symmetry for an exponential reduction in system size. This potential can in practice be thwarted if the symmetry is only "approximate": some permutations in the targeted symmetry group do not leave the model invariant. The work of [7, 5, 18] generalized symmetry reduction to systems where, despite the imprecision in the symmetry, a bisimilar quotient can be constructed. The results in [14, 16] allow in principle arbitrary deviations from symmetry, but the reduction of course dwindles with the divergence from perfect symmetry. Our work, in contrast, deals with a different reason for limited effect of symmetry reduction: an insignificant symmetry group. To the best of our knowledge, our work is the first to apply symmetry reduction based on a *large* (say, the full) group to a model featuring a *small* (say, the rotational) group.

Our notion of safety bisimulation bears some resemblance with that of *weak bisimulation* [12]. The latter relates systems that are bisimilar up to externally unobservable actions, often called τ-transitions. Our setting is in a sense lower-level, as we do not distinguish between visible and invisible system steps and thus do not have τ-transitions.

Unrelated to symmetry, [3] defines an *implementation* relation that compares sets of executions rather than computation trees. In addition, unlike our notions of safety simulation and *bi*simulation, that notion does not seem to generalize to equivalence of structures. In particular, it does not guarantee that if an abstract model fails to satisfy a property, then so does the concrete model.

Future work. We plan to investigate precisely how to detect well-architectedness and architectural symmetry. If a system model is well-architected, it is usually not so by coincidence, but *by design*. For such systems, suspected well-architectedness can perhaps be verified at a high-level abstraction layer, akin to symmetry being verified or enforced at the program text level. As a last resort, well-architectedness can also be verified at the structure level, using a reachability pass from I forward, resulting in a set *Reached*, and one from I backward, resulting in a set $\mathit{Reached}^{-1}$. The structure is well-architected exactly if $\mathit{Reached} \subseteq \mathit{Reached}^{-1}$. Contrast the cost of this check with verifying symmetry, which is *graph-isomorphism complete*.

Regarding architectural symmetry, our approach to detecting it is based on the observation sketched earlier that $\pi(R^+) = R^+$ iff $\pi(R^+) \subseteq R^+$ iff $\pi(R) \subseteq R^+$. That is, a model is architecturally symmetric exactly if every permuted transition can be simulated by a finite-length path.

An open question is how symmetry reduction based on *process counters* [7, 13] can be applied to a system architecturally symmetric with respect to the full symmetry group. Since our approach does not require full symmetry, a translation of the program text as described in [8] is not quite applicable.

Acknowledgments. The authors wish to thank E. Allen Emerson for his inspirational comments on this work, and Georg Weissenbacher for suggesting practical motivations and for revisions on early drafts.

References

1. Krste Asanovic, Ras Bodik, Bryan Christopher Catanzaro, Joseph James Gebis, Parry Husbands, Kurt Keutzer, David Patterson, William Lester Plishker, John Shalf, Samuel Webb Williams, and Katherine Yelick. The landscape of parallel computing research: A view from Berkeley. Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley, 2006.
2. George Candea, James Cutler, and Armando Fox. Improving availability with recursive microreboots: a soft-state system case study. *Performance Evaluation*, 2004.
3. Xiaofang Chen, Steven German, and Ganesh Gopalakrishnan. Transaction based modeling and verification of hardware protocols. In *Formal Methods in Computer-Aided Design (FMCAD)*, 2007.
4. Edmund Clarke, Reinhard Enders, Thomas Filkorn, and Somesh Jha. Exploiting symmetry in temporal logic model checking. *Formal Methods in System Design (FMSD)*, 1996.
5. Allen Emerson, John Havlicek, and Richard Trefler. Virtual symmetry reduction. In *Logic in Computer Science (LICS)*, 2000.
6. Allen Emerson and Prasad Sistla. Symmetry and model checking. *Formal Methods in System Design (FMSD)*, 1996.
7. Allen Emerson and Richard Trefler. From asymmetry to full symmetry: New techniques for symmetry reduction in model checking. In *Correct Hardware Design and Verification Methods (CHARME)*, 1999.
8. Allen Emerson and Thomas Wahl. On combining symmetry reduction and symbolic representation for efficient model checking. In *Correct Hardware Design and Verification Methods (CHARME)*, 2003.
9. The FlexRay Consortium, http://www.flexray.com. *FlexRay—The communication system for advanced automotive control applications*.
10. Günter Heiner and Thomas Thurner. Time-triggered architecture for safety-related distributed real-time systems in transportation systems. In *Fault-Tolerant Computing Symposium (FTCS)*, 1998.
11. Norris Ip and David Dill. Better verification through symmetry. *Formal Methods in System Design (FMSD)*, 1996.
12. Robin Milner. Operational and algebraic semantics of concurrent processes. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*. MIT Press, 1990.
13. Amir Pnueli, Jessie Xu, and Leonore Zuck. Liveness with $(0, 1, \infty)$-counter abstraction. In *Computer-Aided Verification (CAV)*, 2002.
14. Prasad Sistla and Patrice Godefroid. Symmetry and reduced symmetry in model checking. *Transactions on Programming Languages and Systems (TOPLAS)*, 2004.
15. Fabio Somenzi. *The CU Decision Diagram Package, release 2.3.1*. University of Colorado at Boulder, `http://vlsi.colorado.edu/~fabio/CUDD/`.
16. Thomas Wahl. Adaptive symmetry reduction. In *Computer-Aided Verification (CAV)*, 2007.
17. Thomas Wahl, Nicolas Blanc, and Allen Emerson. SVISS: Symbolic verification of symmetric systems. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2008.
18. Ou Wei, Arie Gurfinkel, and Marsha Chechik. Identification and counter abstraction for full virtual symmetry. In *Correct Hardware Design and Verification Methods (CHARME)*, 2005.