# Dimming Relations for the Efficient Analysis of Concurrent Systems via Action Abstraction

Rocco Nicola, Giulio Iacobelli, Mirco Tribastone



# Dimming Relations for the Efficient Analysis of Concurrent Systems via Action Abstraction

Rocco De Nicola[1], Giulio Iacobelli[2], and Mirco Tribastone[2]

[1] IMT — Institute for Advanced Studies Lucca, Italy
rocco.denicola@imtlucca.it
[2] Electronics and Computer Science
University of Southampton, United Kingdom
{g.iacobelli,m.tribastone}@soton.ac.uk

**Abstract.** We study models of concurrency based on labelled transition systems where abstractions are induced by a partition of the action set. We introduce *dimming relations* which are able to relate two models if they can mimic each other by using actions from the same partition block. Moreover, we discuss the necessary requirements for guaranteeing compositional verification. We show how our new relations and results can be exploited when seemingly heterogeneous systems exhibit analogous behaviours manifested via different actions. Dimming relations make the models more homogeneous by collapsing such distinct actions into the same partition block. With our examples, we show how these abstractions may considerably reduce the state-space size, in some cases from exponential to polynomial complexity.

## 1 Introduction

Behavioural relations are powerful techniques to reason about models based on labelled transition systems. The classic notion of bisimulation relates two states $P$ and $Q$ such that any action performed by $P$ can be matched by the same action from $Q$ (and vice versa). Weak bisimulation exploits the fact that some behaviour is abstracted away by hiding the identity of certain actions that can be considered uninteresting at some desired level of detail [1]. Alternatively, in some situations, by exploiting specific structural properties, e.g. symmetries [2], it is possible to study the properties of a given model by considering another one with a smaller state-space size, thus reducing the cost of the analysis.

We propose an approach that lies between ignoring actions and accounting for their similarities. We study behavioural relations that take a coarse-grained view of a model, based on partitioning the set of actions that it can exhibit. Our first notion relates $P$ and $Q$ whenever $Q$ is able to match a given action $a$ performed by $P$ *with any action* in the same partition block of $a$ (and vice versa). We call this *dimmed bisimulation* to highlight that, based on the action partition, the modeller is able to see and reason about the original system at different levels of detail and thus with different levels of accuracy. Clearly, dimmed bisimulation equivalence is less discriminating than bisimilarity, whenever the partition over

the actions is nontrivial. For instance, using standard process algebraic notation, let us consider $a_1.\mathbf{0} + a_2.\mathbf{0}$: a simple process that offers a choice between two actions, $a_1$ and $a_2$, and then stops. This will be dimmed bisimilar to $a_1.\mathbf{0}$ if both $a_1$ and $a_2$ belong to the same partition block; evidently, $a_1.\mathbf{0}+a_2.\mathbf{0}$ is not bisimilar to $a_1.\mathbf{0}$. This simple example shows that it is possible to obtain a more compact description when the modeller is content with reasoning at the level of partition representatives instead of considering detailed concrete actions. Indeed, we will present more substantial examples where the effect of this abstraction will be more significant. To provide some intuition behind this approach, let $\|$ denote a generic parallel operator and consider $a_1.\mathbf{0} \parallel a_2.\mathbf{0} \parallel \cdots \parallel a_n.\mathbf{0}$, e.g., a model of $n$ independent threads of execution, each performing a distinct computation $a_i$, $1 \leq i \leq n$, and then stopping. The state space size of this (concrete) system is $2^n$; however, allowing all $a_i$ to be actions of the same partition block permits a dramatic reduction of the (abstract) state space to only $n+1$ states, where each of them tracks the number of processes that have not become inert ($\mathbf{0}$) yet.

An analogous state-space reduction could have been obtained by assuming that the actions in the same partition block represent unnecessary detail in the model. Thus, in $a_1.\mathbf{0} + a_2.\mathbf{0}$ one would replace $a_1$ and $a_2$ with *internal* $\tau$ actions, yielding $\tau.\mathbf{0}+\tau.\mathbf{0}$, which is bisimilar to $\tau.\mathbf{0}$. However, there are two major drawbacks with this approach. Firstly, in such a reduced form it is not possible to keep track of the concrete actions from which the internal ones are originated. Secondly, sometimes this abstraction may not be possible, for instance when $a_1$ and $a_2$ are intended to be used for synchronising with other processes.

More closely related to our behavioural relations is turning one action, say $a_2$, into another one. For instance, identification of $a_1.\mathbf{0}+a_2.\mathbf{0}$ with $a_1.\mathbf{0}$ can be done by using $(a_1.\mathbf{0}+a_2.\mathbf{0})[f]$ where $f$ is the relabelling function such that $f(a_1) = a_1$ and $f(a_2) = a_1$. Indeed, we provide a characterisation of dimmed bisimulation based on the existence of an appropriate $f$ such that $P[f]$ is bisimilar, in the classical sense, to $Q[f]$. Thus, dimmed bisimulation is able to relate processes that behave bisimilarly after an appropriate relabelling of their actions.

A highly desirable property of any behavioural relation for process algebra is establishing that it is preserved by its operators. Here we study a CSP-like process algebra (see, e.g., [3]) where the parallel operator is parameterised by a synchronisation action set. We find that dimmed bisimilarity is preserved only if certain syntactic conditions are met. For instance, it is preserved by parallel composition under *singleton coherence*, i.e., if non-trivial partition blocks contain only actions that are performed independently and not used for synchronisation.

Unfortunately, the singleton coherence restriction rules out the possibility of compositional reasoning for models of practical interest. For instance, using again the simple example discussed above, $a_1.\mathbf{0} + a_2.\mathbf{0}$ could not be placed in any context where both $a_1$ and $a_2$ are synchronisation actions. However, this represents a typical modelling pattern whereby a process is able to offer two (or more) different options to the environment. In order to deal with situations of this kind, we introduce *dimmed simulation* (whereby, analogously to the classic notion of simulation, any action performed by $P$ can be matched by $Q$ with

any other action in the same partition block, but not vice versa). We will see that *dimmed similarity* is preserved by parallel composition under the much more generous assumption of *block coherence*, essentially requiring that a whole partition block must be included in a synchronisation set if this contains an action of the partition block.

We will apply our results to a number of models which prototypically show typical patterns of communication in distributed systems, where *heterogeneity* is expressed as analogous behaviour with distinct actions. We will study a simple model of fork/join mechanism. This can be relevant, for instance, to capture certain aspects of novel programming paradigms such as MapReduce (see, e.g., [4]). Here, a distinct action type is used for each worker thread to signal that it has finished work. Second, we study a producer/consumer system mediated by a buffer (e.g., [5]), where the existence of different classes of items is captured by having the buffer expose distinct pairs of *put* and *get* actions for each item class. Finally, we consider Milner's *cyclers* model [1], a benchmark in the process algebra literature. In all these cases, the state-space size of such models grows exponentially with the number of components in the model. Our dimming relations allow us to analyse systems after making them more *homogeneous*, by offering a coarser-grained view, as induced by action partitioning. Obviously, the modeller has to be satisfied by the provided abstract view. For example, she may be content with ensuring that the fork/join model only captures that *some* sub-tasks have been fulfilled by *some* threads; or that an action *put* enables an action *get* in any buffer place, without necessarily wanting to know *which* specific thread has completed or which specific item class has been handled. In some of our examples, it turns out that it is possible to obtain dimmed (bi-)similar processes with state-space sizes of polynomial, rather than exponential, complexity.

*Paper outline.* After introducing our CSP-like process algebra, Section 2 discusses dimmed bisimulation and studies compositionality and its characterisation in terms of action relabelling. Section 3 applies our results to the case studies. Section 4 introduces dimmed simulation. Section 5 discusses related work, while Section 6 briefly concludes. Unless otherwise stated, all proofs can be found in the technical report available at http://eprints.imtlucca.it/1655/.

## 2 Dimmed Bisimulation

We carry out our investigation in the context of a process algebra with CSP-style semantics. However, with appropriate changes our ideas of dimmed relations carry over to other synchronisation operators such as the binary one in CCS [1].

**Definition 1 (Process Algebra Syntax).** *Let $\mathcal{A}$ be a set of actions and $\mathcal{L} = \mathcal{A} \cup \{\tau\}$, with $\tau \notin \mathcal{A}$, be the set of labels. Our process algebra has the syntax*

$$P ::= \mathbf{0} \mid a.P \mid P + P \mid K \mid P[f] \mid P \parallel_L P \mid P/L$$

*where $a \in \mathcal{L}$, $K \in \mathcal{K}$, with $\mathcal{K}$ the set of constants and $K \triangleq P$, $f : \mathcal{L} \to \mathcal{L}$ is a relabelling function with $f(\tau) = \tau$, and $L \subseteq \mathcal{A}$. Let $\mathcal{P}$ be the set of processes, that is the set of terms generated by the grammar above.*

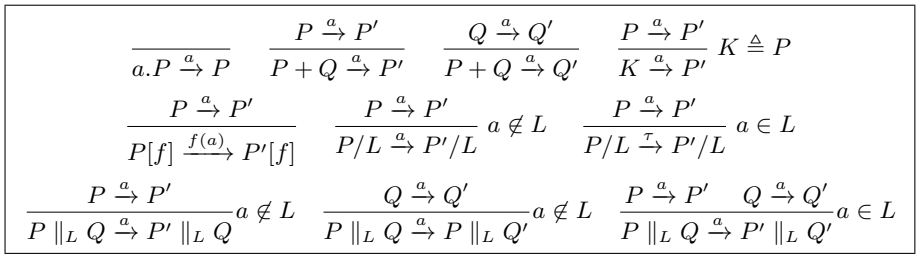


**Fig. 1.** Process algebra semantics.

We use standard syntax, thus: $\tau$ is the internal action; $\mathbf{0}$ is the inert process, that does nothing; $a.P$ denotes *prefixing*, which can perform an $a$-action and become $P$; $P + P$ offers a *choice* between behaviours; $K$ is a *constant*, for recursion; $P[f]$ is a process to which a *relabelling* of its actions is applied (where $\tau$ cannot be relabelled); $P \parallel_L P$ is the generalised *parallel* operator, whereby the two operands are required to synchronise only over the actions that are in the set $L$; $P/L$ models *hiding*, whereby an action performed by $P$ is made internal if it is in $L$.[1] These rules are captured by the operational semantics in Fig. 1.

**Notation 1** *Throughout the paper, we let $\mathfrak{F} = \{\mathcal{F}_1, \ldots, \mathcal{F}_m\}$ denote a partition of $\mathcal{A}$. If $a \in \mathcal{A}$ we let $[a]_\mathfrak{F}$ denote the partition block $\mathcal{F}_i$ such that $a \in \mathcal{F}_i$; when $\mathfrak{F}$ is clear from the context, we omit the subscript and simply write $[a]$. We extend the notation $[\cdot]$ to $\tau$ by setting $[\tau] = \{\tau\}$.*

**Definition 2.** *Let $\mathfrak{F}$ be a partition of $\mathcal{A}$. We say that $P \xrightarrow{[a]} P'$ iff there exists $b \in [a]$ such that $P \xrightarrow{b} P'$.*

According to this definition and to the previous notation, if $P \xrightarrow{\tau} P'$ then we write $P \xrightarrow{[\tau]} P'$. Intuitively, Definition 2 gives us the *dimmed behaviour* of a process $P$: if it can make a *concrete* action $b$ then we say that it can make an *abstract* action $[a]$, which essentially stands for saying that "$P$ can make *at least one* of the actions of the partition block to which $b$ belongs." Our notion of dimmed bisimulation allows processes to match each other's actions so long as they are in the same partition block.

**Definition 3 (Dimmed Bisimulation).** *Given a partition $\mathfrak{F}$, a binary relation $\mathcal{R}$ over $\mathcal{P}$ is an* $\mathfrak{F}$-dimmed bisimulation *iff whenever $(P, Q) \in \mathcal{R}$ and $a \in \mathcal{L}$:*

- *if $P \xrightarrow{[a]} P'$ then $Q \xrightarrow{[a]} Q'$ and $(P', Q') \in \mathcal{R}$;*
- *if $Q \xrightarrow{[a]} Q'$ then $P \xrightarrow{[a]} P'$ and $(P', Q') \in \mathcal{R}$.*

*Two processes $P, Q$ are* $\mathfrak{F}$-dimmed bisimilar*, written $P \sim_\mathfrak{F} Q$, iff there exists an $\mathfrak{F}$-dimmed bisimulation that relates them.*

[1] Our notions naturally extend to weak variants.

Let us stress that the classic notion of bisimulation, hereafter denoted by $\sim$, can be recovered by choosing $\mathfrak{F}$ as the trivial singleton partition (which yields $[a] = \{a\}$ for all $a \in \mathcal{A}$). With an appropriate choice of the action partition, it is possible to find dimmed bisimilar processes that are easier to analyse. For instance, using again the simple process presented in Section 1, it holds that $a_1.\mathbf{0} + a_2.\mathbf{0} \sim_{\mathfrak{F}} a_1.\mathbf{0}$ (and also that $a_1.\mathbf{0} + a_2.\mathbf{0} \sim_{\mathfrak{F}} a_2.\mathbf{0}$) if $\{a_1, a_2\} \in \mathfrak{F}$. This is because $\mathcal{R} = \{(a_1.\mathbf{0} + a_2.\mathbf{0}, a_1.\mathbf{0}), (\mathbf{0}, \mathbf{0})\}$ is an $\mathfrak{F}$-dimmed bisimulation. For instance, both $a_1.\mathbf{0}+a_2.\mathbf{0} \xrightarrow{a_1} \mathbf{0}$ and $a_1.\mathbf{0}+a_2.\mathbf{0} \xrightarrow{a_2} \mathbf{0}$ imply that $a_1.\mathbf{0}+a_2.\mathbf{0} \xrightarrow{[a_1]} \mathbf{0}$ and $a_1.\mathbf{0} \xrightarrow{[a_1]} \mathbf{0}$ since $a_1.\mathbf{0} \xrightarrow{a_1} \mathbf{0}$. Please notice that, in this particular case, the dimmed bisimilar process has the same number of states, but fewer transitions. Later we will provide examples where also the number of states will be reduced.

As expected, similarly to classical bisimulation, the following holds.

**Theorem 1.** *For any partition $\mathfrak{F}$, the relation $\sim_{\mathfrak{F}}$*

*a) is an equivalence relation;*
*b) is the largest $\mathfrak{F}$-dimmed bisimulation;*
*c) satisfies the following property: $P \sim_{\mathfrak{F}} Q$ iff, for any $a \in \mathcal{L}$,*
- *if $P \xrightarrow{[a]} P'$ then $Q \xrightarrow{[a]} Q'$ and $P' \sim_{\mathfrak{F}} Q'$;*
- *if $Q \xrightarrow{[a]} Q'$ then $P \xrightarrow{[a]} P'$ and $P' \sim_{\mathfrak{F}} Q'$.*

Let us observe that, with $\{a_1, a_2\} \in \mathfrak{F}$, it is also possible to establish the relation $a_1.\mathbf{0} + a_2.\mathbf{0} \sim_{\mathfrak{F}} a_1.\mathbf{0} + a_1.\mathbf{0}$. This intuitively suggests that dimmed bisimulation can be understood as a form of relabelling that turns actions of a partition block into possibly distinct actions within the same partition block. Indeed, in this simple case we have that $(a_1.\mathbf{0} + a_2.\mathbf{0})[f] \sim (a_1.\mathbf{0} + a_1.\mathbf{0})[f]$ for $f$ such that $f(a_1) = a_1$ and $f(a_2) = a_1$.

Dimmed bisimulation can be characterised in terms of actions relabelling. We start with identifying functions that relabel within the same block of $\mathfrak{F}$.

**Definition 4.** *A relabelling function $f : \mathcal{L} \to \mathcal{L}$ is said to be* partition-preserving *(pp) for $\mathfrak{F}$ iff for each $\mathcal{F} \in \mathfrak{F}$ and for each $a \in \mathcal{F}$ it holds that $f(a) \in \mathcal{F}$.*

**Theorem 2 (Characterisation of Dimmed Bisimulation via relabelling).** *Given a partition $\mathfrak{F}$, $P \sim_{\mathfrak{F}} Q$ iff there exists a pp-function $f$ for $\mathfrak{F}$ such that $P[f] \sim Q[f]$.*

We now study whether dimmed bisimulation is preserved by the operators of our process algebra. For hiding and parallel composition, that are parameterised by an action set $L$, we need to impose syntactic restrictions on the set $L$, which we call *block coherence* and *singleton coherence*, respectively. Roughly speaking, block coherence requires that an element of $\mathfrak{F}$ is either completely in the action set $L$, or completely outside. Singleton coherence, instead, requires that each action belonging to the action set $L$ cannot be aggregated with other actions.

**Definition 5.** *Given a partition $\mathfrak{F}$ and $L \subseteq \mathcal{A}$, we say that $L$ is*

- block coherent *with $\mathfrak{F}$ iff $\forall \mathcal{F} \in \mathfrak{F}$ such that $\mathcal{F} \cap L \neq \emptyset$ we have $\mathcal{F} \subseteq L$ .*

– singleton coherent *with* $\mathfrak{F}$ *iff* $\forall \mathcal{F} \in \mathfrak{F}$ *such that* $\mathcal{F} \cap L \neq \emptyset$ *we have* $|\mathcal{F}| = 1$ .

For instance, let $\mathfrak{F} = \big\{\{a_1, a_2\}, \{b\}\big\}$. Then $\{b\}$ is singleton coherent with $\mathfrak{F}$, whereas $\{a_1, a_2\}$ is not. However, $\{a_1, a_2\}$ is block coherent with $\mathfrak{F}$. Furthermore, every action set $L$ is singleton coherent with the trivial singleton partition of the action set; finally, the empty set is singleton coherent with any $\mathfrak{F}$.

**Theorem 3 (Compositionality for Dimmed Bisimulation).** *Let $P$ and $Q$ be two processes such that $P \sim_{\mathfrak{F}} Q$. Then it holds that:*

*i)* $a.P \sim_{\mathfrak{F}} b.Q$ *for any* $a, b$ *such that* $[a] = [b]$*;*
*ii)* $P + R \sim_{\mathfrak{F}} Q + S$ *for any two processes* $R$ *and* $S$ *such that* $R \sim_{\mathfrak{F}} S$*;*
*iii)* $P[g] \sim_{\mathfrak{F}} Q[g]$ *for any function* $g$ *that is partition preserving for* $\mathfrak{F}$*;*
*iv)* $P/L \sim_{\mathfrak{F}} Q/L$ *for* $L$ *block coherent with* $\mathfrak{F}$*;*
*v)* $P \parallel_L R \sim_{\mathfrak{F}} Q \parallel_L S$ *if* $R \sim_{\mathfrak{F}} S$ *and* $L$ *is singleton coherent with* $\mathfrak{F}$*.*

In general, dimmed bisimulation is not preserved if the above syntactical restrictions are not satisfied. To see this, let us take, for instance, processes $a_1.\mathbf{0}$ and $a_2.\mathbf{0}$, and some partition $\mathfrak{F}$ such that $\{a_1, a_2\} \subseteq [a_1]$. Then it holds, that $a_1.\mathbf{0} \sim_{\mathfrak{F}} a_2.\mathbf{0}$. For relabelling, let $b_1 \notin [a_1]$ and define $f$ such as $f(a_1) = b_1$, $f(a_2) = a_2$, and $f(b_1) = b_1$, whence $f$ is not a pp-function. Then, it holds that $a_1.\mathbf{0}[f] \not\sim_{\mathfrak{F}} a_2.\mathbf{0}[f]$. For hiding, $a_1.\mathbf{0}/L \not\sim_{\mathfrak{F}} a_2.\mathbf{0}/L$ if $L = \{a_1\}$. Finally, dimmed bisimulation is not preserved by parallel composition if the action set is not singleton coherent with $\mathfrak{F}$. For instance, $a_1.\mathbf{0} \parallel_{\{a_1\}} a_1.\mathbf{0} \not\sim_{\mathfrak{F}} a_2.\mathbf{0} \parallel_{\{a_1\}} a_1.\mathbf{0}$.

*Working with dimmed bisimilar candidates.* The following fact establishes a clear relationship between a process $P$ and the process obtained by inserting $P$ in a context making use of a relabelling pp-function. In practice, it allows us to find a nontrivial (i.e., nonidentical) candidate bisimulating process.

**Proposition 1.** *Let $\mathfrak{F}$ be a partition and $f$ be a pp-function for $\mathfrak{F}$. Then it holds that $P \sim_{\mathfrak{F}} P[f]$.*

*Proof.* The relation $\{(P', P'[f]) \mid f \text{ is a pp-function for } \mathfrak{F}\}$ is an $\mathfrak{F}$-dimmed bisimulation. □

Notice that, in general, the state space of $P$ is as large as that of $P[f]$. However, in some cases, it may be easier to find a smaller process by studying how the pp-function $f$ distributes over the process algebra operators.

**Proposition 2.** *Let $f$ be a pp-function for $\mathfrak{F}$. It holds that*

*i)* $(P[g])[f] \sim_{\mathfrak{F}} P[f]$ *for any function* $g$ *that is partition preserving for* $\mathfrak{F}$*;*
*ii)* $(P/L)[f] \sim_{\mathfrak{F}} (P[f])/L$ *for* $L$ *block coherent with* $\mathfrak{F}$*;*
*iii)* $(P \parallel_L Q)[f] \sim_{\mathfrak{F}} P[f] \parallel_L Q[f]$ *if* $L$ *is singleton coherent with* $\mathfrak{F}$*.*

Interestingly, dimmed bisimilarity behaves rather differently than bisimilarity with respect to distributivity. For instance, in *i)* the information of the pp-function $g$ is lost, while bisimilarity uses function composition $f \circ g$; the analogous

statement to *ii)* for bisimilarity requires a similar form of coherence for $L$, i.e. $a \in L$ iff $f(a) \in L$; finally, *iii)* uses a weaker assumption on $f$ than bisimilarity, for which $f$ must be injective, but has a syntactic restriction on the synchronisation set, unlike bisimilarity. In fact, singleton coherence and partition preservation coincide with requiring that $f$ be injective on the synchronised actions.

Let us also remark that, in general, distributivity may not be preserved when the side conditions are not satisfied. For instance:

- $(a_1.\mathbf{0}[g])[f] \not\sim_{\mathfrak{F}} (a_1.\mathbf{0})[f]$ if $g(a_1) = b$, with $b \notin [a_1]$, and $f(a_1) = a_1$ and $f(b) = b$;
- $\big((a_1.\mathbf{0}+a_2.\mathbf{0})/\{a_1\}\big)[f] \not\sim_{\mathfrak{F}} \big((a_1.\mathbf{0}+a_2.\mathbf{0})[f]\big)/\{a_1\}$ if $f(a_2) = a_1$ and $f(a_1) = a_1$, with $\{a_1, a_2\} \in \mathfrak{F}$;
- $(a_1.\mathbf{0} \parallel_{\{a_1,a_2\}} a_2.\mathbf{0})[f] \not\sim_{\mathfrak{F}} (a_1.\mathbf{0})[f] \parallel_{\{a_1,a_2\}} (a_2.\mathbf{0})[f]$, with the same $f$ as above.

*Different levels of dimming.* So far, all our results have assumed a given fixed partition of the action set. Now we turn to considering what can be said for models with different levels of dimming, induced by different partitions. In general, for any two partitions, establishing dimmed bisimilarity with one partition does not allow us to infer dimmed bisimilarity for the other. For example, $a_1.\mathbf{0}+a_2.\mathbf{0}+b.\mathbf{0} \sim_{\mathfrak{F}_1} a_1.\mathbf{0}+a_2.\mathbf{0}$ for $\mathfrak{F}_1 = \big\{\{a_1, b\}, \{a_2\}\big\}$; but $a_1.\mathbf{0}+a_2.\mathbf{0}+b.\mathbf{0} \not\sim_{\mathfrak{F}_2} a_1.\mathbf{0} + a_2.\mathbf{0}$ if $\mathfrak{F}_2 = \big\{\{a_1, a_2\}, \{b\}\big\}$. However, it turns out that the usual partial order based on *partition refinement* captures the intuitive idea that one partition provides a higher level of abstraction than the other. Formally, given two partitions $\mathfrak{F}_1$ and $\mathfrak{F}_2$, one says that $\mathfrak{F}_1$ is a *refinement* of $\mathfrak{F}_2$, written as $\mathfrak{F}_1 \leq \mathfrak{F}_2$, if every element of $\mathfrak{F}_1$ is a subset of an element of $\mathfrak{F}_2$. In this case, it is also said that $\mathfrak{F}_1$ is *finer* than $\mathfrak{F}_2$ and that $\mathfrak{F}_2$ is *coarser* than $\mathfrak{F}_1$.

**Proposition 3.** *Let $\mathfrak{F}_1$, $\mathfrak{F}_2$ be two partitions of $\mathcal{A}$ such that $\mathfrak{F}_1 \leq \mathfrak{F}_2$ and let $P$ and $Q$ be two processes such that $P \sim_{\mathfrak{F}_1} Q$; then it holds that $P \sim_{\mathfrak{F}_2} Q$.*

*Proof.* $\mathfrak{F}_1 \leq \mathfrak{F}_2$ entails that $[a]_{\mathfrak{F}_1} \subseteq [a]_{\mathfrak{F}_2}$ for any $a \in \mathcal{A}$. Thus $\mathcal{R} \triangleq \{(P', Q') \mid P' \sim_{\mathfrak{F}_1} Q'\}$ contains $(P, Q)$ and is an $\mathfrak{F}_2$-dimmed bisimulation. □

Let us denote by $\mathfrak{F}_0$ the trivial singleton partition, for which it holds that $\mathfrak{F}_0 \leq \mathfrak{F}$ for any $\mathfrak{F}$. Since $\sim = \sim_{\mathfrak{F}_0}$, as a corollary of the above proposition we have the following.

**Corollary 1.** *$\sim$ implies $\sim_{\mathfrak{F}}$ for any partition $\mathfrak{F}$.*

## 3 Dimmed Bisimulation at Work

In this section we present how to exploit dimmed bisimulation in three examples of typical modelling patterns of distributed systems. In all cases, Corollary 1 will be used in order to find dimmed bisimilar processes that are much easier to analyse than the original ones. Given some process $P$, the idea is to first find some $Q$ such that $P \sim_{\mathfrak{F}} Q$; then, using well-known algorithms [6], one reduces

$Q$ up to bisimulation into $R$, from which it holds that $P \sim_{\mathfrak{F}} R$. For instance, the relation $a_1.\mathbf{0} + a_2.\mathbf{0} \sim_{\mathfrak{F}} a_1.\mathbf{0}$ can be interpreted as establishing first that $a_1.\mathbf{0} + a_2.\mathbf{0} \sim_{\mathfrak{F}} a_1.\mathbf{0} + a_1.\mathbf{0}$, and then observing that $a_1.\mathbf{0} + a_1.\mathbf{0} \sim a_1.\mathbf{0}$.

Our first example studies a concurrent system with fork/join synchronisation, which could be used for a high-level model of a MapReduce computing task [4].

*Example 1 (A Fork/Join System).* Let us consider $\mathcal{A} = \{fork, join\} \cup \{w_i \mid 1 \leq i \leq n\}$. Our model consists of a master process, denoted by $F$, which invokes (*fork*) $n$ worker threads. Each worker thread, $W_i$, performs a distinct type of computation, modelled as a distinct action $w_i$. Once all threads finish their task, the master process collects the results (*join*) and repeats the cycle invoking the threads again. The model is as follows.

$$F \triangleq fork.join.F \qquad W_i \triangleq fork.w_i.join.W_i \qquad 1 \leq i \leq n.$$
$$M := F \parallel_L W_1 \parallel_L \cdots \parallel_L W_n, \qquad L = \{fork, join\}.$$

Model $M$ has a state space size that grows as $2^n + 1$. In addition, it cannot be further minimised up to bisimulation. Dimmed bisimulation, on the other hand, may yield a simpler model which enjoys linear complexity of the state space size. To show this, let us take $\mathfrak{F} = \big\{\{fork\}, \{join\}, \{w_i \mid 1 \leq i \leq n\}\big\}$; that is, let us assume that the modeller wishes to abstract away from the identity of the worker thread that finishes works, but wants nevertheless to observe that *some thread* has completed. Then $L$ is singleton coherent with $\mathfrak{F}$, and we have that $W_i \sim_{\mathfrak{F}} W_1$ for all $1 \leq i \leq n$. Thus, Theorem 3 yields that $M \sim_{\mathfrak{F}} \bar{M}$, with

$$\bar{M} := F \parallel_L \underbrace{W_1 \parallel_L \cdots \parallel_L W_1}_{n \text{ times}}.$$

Now, $\bar{M}$ still has $2^n + 1$ states, but *it can* be minimised up to bisimulation due to the symmetry among the worker threads, which are now copies of the same process. In this case, it is sufficient to just *count* how many worker threads are performing $w_1$. More precisely, let us consider the process $\bar{W}_0$ defined as

$$\bar{W}_0 \triangleq fork.\bar{W}_1, \qquad \bar{W}_i \triangleq w_1.\bar{W}_{i+1}, \quad \text{for } 1 \leq i \leq n, \qquad \bar{W}_{n+1} \triangleq join.\bar{W}_0.$$

Then it holds that $\underbrace{W_1 \parallel_L \cdots \parallel_L W_1}_{n \text{ times}} \sim \bar{W}_0$, from which, by Proposition 1, it follows that $M \sim_{\mathfrak{F}} F \parallel_L \bar{W}_0$. Hence, given a model with $2^n + 1$ states, we were able to construct a dimmed bisimilar one with only $n + 2$ states.

In the above example, nontrivial parts of the action set involve actions which are never synchronised. That is, dimmed bisimulation can be inferred *compositionally*, starting from the simplest concurrent processes, because they are composed together over synchronisation action sets that satisfy singleton coherence. By contrast, the following examples can still be related to more compact dimmed bisimilar processes; however, dimmed bisimilarity *cannot* be inferred compositionally because the synchronisation sets are not singleton coherent. Thus a

| $n_1$ | $n_2$ | $m_1$ | $m_2$ | $k$ | $\lvert M\rvert$ | $\lvert\bar{M}\rvert$ | $\lvert M\rvert/\lvert\bar{M}\rvert$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 48 | 18 | 2.67 |
| 2 | 2 | 2 | 2 | 1 | 243 | 50 | 5.06 |
| 1 | 1 | 1 | 1 | 10 | 1056 | 99 | 10.67 |
| 2 | 2 | 2 | 2 | 10 | 5346 | 275 | 19.44 |
| 1 | 1 | 1 | 1 | 100 | 82416 | 909 | 90.67 |
| 2 | 2 | 2 | 2 | 100 | 417213 | 2525 | 165.23 |
| 3 | 3 | 2 | 2 | 100 | 741744 | 3535 | 209.83 |

**Table 1.** State-space sizes for $M$ and $\bar{M}$ in Example 2 (after minimisation of both processes up to bisimulation), denoted by $|M|$ and $|\bar{M}|$, respectively.

relation must be directly given for the whole composite process. In Section 4 we will, however, show how compositional reasoning on the same examples can be recovered at the cost of establishing only dimmed simulation.

Our second case study is a model of a producer/consumer system where the interaction is mediated by a buffer of finite capacity (see also [5]).

*Example 2 (Multi-class Producer/Consumer).* For simplicity, let us consider two classes of producers and two classes of users which share the same buffer. Let $m_1$ (resp., $m_2$) be the number of producers of the first (resp., second) class; let $n_1$ and $n_2$ be the number of consumers. Finally, let $k$ be the buffer capacity.

A class-$i$ producer, for $i = 1, 2$, is modelled by $P_i \triangleq \mathit{prod}_i.\mathit{put}_i.P_i$; here $\mathit{prod}_i$ models an independent action that describes the production of an item, whereas $\mathit{put}_i$ is a synchronisation action to be performed with the buffer: It can be executed only when there is at least one place available in the buffer. In a similar fashion, the model of a class-$i$ consumer is given by $C_i \triangleq \mathit{get}_i.\mathit{cons}_i.C_i$. In this case, $\mathit{get}_i$ is a synchronisation action with the buffer, whereas the (independent) consumption of an item is modelled by action $\mathit{cons}_i$. Finally, a place in the buffer is $B \triangleq \mathit{put}_1.\mathit{get}_1.B + \mathit{put}_2.\mathit{get}_2.B$. Our overall model is

$$M := \big(P_1[m_1] \parallel_\emptyset P_2[m_2]\big) \parallel_{L_1} B[k] \parallel_{L_2} \big(C_1[n_1] \parallel_\emptyset C_2[n_2]\big)$$

where $L_1 = \{\mathit{put}_1, \mathit{put}_2\}$, $L_2 = \{\mathit{get}_1, \mathit{get}_2\}$, and $S[l]$ abbreviates $(\underbrace{S \parallel_\emptyset \cdots \parallel_\emptyset S}_{l \text{ times}})$.

A reasonable abstraction is not to insist on keeping the two classes of producers (resp. consumers) distinct. Let us thus consider the action partition $\mathfrak{F} = \big\{L_1, L_2, \{\mathit{prod}_1, \mathit{prod}_2\}, \{\mathit{cons}_1, \mathit{cons}_2\}\big\}$. It is possible to show that

$$M \sim_{\mathfrak{F}} \bar{M}, \qquad \text{with} \quad \bar{M} := P_1[m_1 + m_2] \parallel_{L_1} B_1[k] \parallel_{L_2} C_1[n_1 + n_2].$$

That is, producers and consumers of the second class are dimmed bisimilar to those of the first class; furthermore, a buffer place with two distinct actions is dimmed bisimilar to a buffer with actions of a single representative type. As with Example 1, the state-space reduction achieved can be significant. Indeed,

| $n$ | $\lvert M_n\rvert$ | $\lvert\bar{M}_n\rvert$ | $\lvert M_n\rvert/\lvert\bar{M}_n\rvert$ |
|---|---|---|---|
| 3 | 37 | 35 | 1.05 |
| 9 | 6913 | 715 | 9.67 |
| 12 | 73729 | 1820 | 40.51 |
| 15 | 737281 | 3876 | 190.22 |
| 16 | 1572865 | 4845 | 324.64 |
| 17 | 3342337 | 5985 | 558.45 |

**Table 2.** State-space sizes for $M_n$ and $\bar{M}_n$ in Example 3 (after minimisation of both processes up to bisimulation), denoted by $|M_n|$ and $|\bar{M}_n|$, respectively.

Table 1 compares the state-space sizes of $M$ and $\bar{M}$ (after minimisation of both processes up to bisimulation) for different values of $n_1$, $n_2$, $m_1$, $m_2$, and $k$.[2]

Next we discuss a classical example in the process algebra literature.

*Example 3 (Milner's Cyclers [1]).* Milner's cyclers is a model of a scheduler that allows a set of processes $P_1, P_2, \ldots, P_n$ to cyclically perform local computations in succession; that is, process $P_i$ cannot start its computation until $P_{i-1}$ has instructed it to do so. The model constants are given by:

$$P_i \triangleq \gamma_i.Q_i + \sum_{j=1, j\neq i}^{n} \gamma_j.P_i \qquad Q_i \triangleq \alpha_i.R_i \qquad R_i \triangleq \gamma_{s_i}.S_i + \beta_i.T_i$$

$$S_i \triangleq \beta_i.P_i + \sum_{j=1, j\neq i, s_i}^{n} \gamma_j.S_i \qquad T_i \triangleq \gamma_{s_i}.P_i \qquad 1 \leq i \leq n,$$

were $s_i$ denotes the successor of process $i$, i.e., $s_i := i + 1$ if $1 \leq i \leq n - 1$, and $s_i := 1$ if $i = n$. Action $\gamma_i$ represents the signal that process $i$ is able to start the computation; its performance is modelled by action $\alpha_i$; upon completion, the process may signal the start to its successor (hence, to achieve cyclic behaviour $R_n$ will perform action $\gamma_1$), or notify the end of the local computation via $\beta_i$, in either order. Process $S_i$ describes the state of a cycler which has already started the computation and let the next cycler go. In that state, it may witness some $\gamma_j$-action performed by other cyclers; in this case, the cycler ignores this signal and behaves as $S_i$ again. The model is completed by a scheduler, $Sc$, that will enforce the start of $P_1$:

$$Sc \triangleq \gamma_1.Sc' \qquad\qquad Sc' \triangleq \sum_{i=1}^{n} \gamma_i.Sc'$$

Thus, the overall system is described by $M_n := Sc \parallel_L P_1 \parallel_L \cdots \parallel_L P_n$ with $L = \{\gamma_1, \ldots, \gamma_n\}$.

[2] For the derivation of the state spaces of the examples presented in this paper, we used a software tool for a stochastic process algebra [7], which yields a doubly labelled transition system of which we ignored the rate information.

Let us consider the abstraction whereby the modeller is not interested in the specific identity (i.e., the position in the cycle) of a process performing an action. That is, let us take $\mathfrak{F} = \big\{\{\alpha_1, \ldots, \alpha_n\}, \{\beta_1, \ldots, \beta_n\}, \{\gamma_1, \ldots, \gamma_n\}\big\}$ and processes

$$\bar{P} \triangleq \gamma_1.\bar{P} + \gamma_1.\bar{Q} \qquad \bar{Q} \triangleq \alpha_1.\bar{R} \qquad \bar{R} \triangleq \gamma_1.\bar{S} + \beta_1.\bar{T}$$
$$\bar{S} \triangleq \beta_1.\bar{P} + \gamma_1.\bar{S} \qquad \bar{T} \triangleq \gamma_1.\bar{P} \qquad \bar{Sc} \triangleq \gamma_1.\bar{Sc}$$

Let $\bar{M}_n := \bar{Sc} \parallel_L \overbrace{\bar{P} \parallel_L \cdots \parallel_L \bar{P}}^{n \text{ times}}$. Although with this abstraction we loose the circular order, we preserve the feature that local computations must be performed exclusively, i.e., in any state of the system at most one process can perform the corresponding $\alpha$ action. Then, similarly to Example 2, it can be shown that $M_n \sim_{\mathfrak{F}} \bar{M}_n$. Now, $\bar{M}_n$ can be reduced by exploiting the symmetry between the $\bar{P}$ components. The state-space size of Milner's model cannot be reduced up to strong bisimilarity, unlike $\bar{M}_n$ which has polynomial complexity growing as $\binom{n+4}{n}$. Table 2 compares the state spaces for different values of $n$.

## 4 Dimmed Simulation

As mentioned, single coherence may be an impediment to compositional reasoning for some interesting models such as Examples 2 and 3. Fortunately, compositional reasoning can still be applied if one uses dimmed *simulation* instead of dimmed bisimulation. In passing, we note that simulation was introduced by Milner much earlier than bisimulation [8], and it has since then been used for establishing interesting properties of systems. For example, in [9] it is argued that "in many cases, neither trace equivalence nor bisimilarity, but similarity is the appropriate abstraction for computer-aided verification."

**Definition 6 (Dimmed Simulation).** *Given a partition $\mathfrak{F}$, a binary relation $\mathcal{R}$ over $\mathcal{P}$ is an* $\mathfrak{F}$-dimmed simulation *iff whenever $(P, Q) \in \mathcal{R}$ and $a \in \mathcal{L}$:*

- *if $P \xrightarrow{[a]} P'$ then $Q \xrightarrow{[a]} Q'$ and $(P', Q') \in \mathcal{R}$.*

*For two processes $P$ and $Q$, we say that $Q$* $\mathfrak{F}$-dimmedly simulates *$P$, written $P \preceq_{\mathfrak{F}} Q$, if there is an* $\mathfrak{F}$-dimmed simulation *which relates them.*

Similarly to dimmed bisimulation, using the trivial singleton partition we recover the standard notion of simulation between processes, hereafter denoted by $\preceq$. We state the next proposition without proof.

**Proposition 4.** *The following hold:*

*i) $\preceq_{\mathfrak{F}}$ is a preorder;*
*ii) $P \sim_{\mathfrak{F}} Q \implies P \preceq_{\mathfrak{F}} Q \ \wedge \ Q \preceq_{\mathfrak{F}} P$.*

*Mutatis mutandis*, the relationship between dimmed simulation and simulation is the same as the relationship between dimmed bisimulation and bisimulation as discussed in Section 2. That is, Theorem 2 carries over. Further, dimmed

similarity is preserved by partition refinement (cf. Proposition 3), thus similarity implies dimmed similarity (cf. Corollary 1). Propositions 1 and 2 hold also for dimmed simulation by point ii) of Proposition 4.

The results of Theorem 3 would carry over as well. However, in the case of dimmed simulation it is possible to relax the assumption on singleton coherency, which is needed for the preservation of dimmed bisimulation by parallel composition. Here, instead, we will just require block coherency, as well as a form of *homogeneity* of the two operands of the parallel composition with respect to the synchronisation set. To formally define this notion, we denote by $Act(P)$ the set of all actions that can be performed by process $P$; for all $a \in \mathcal{L}$,

$$a \in Act(P) \text{ iff } \exists n \geq 1 : P \xrightarrow{a_1} P_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} P_n \ , \ a_n = a.$$

**Definition 7 (Homogeneous Processes).** *Let $P$ and $Q$ be two processes, $L \subseteq \mathcal{A}$, and $\mathfrak{F}$ a partition. $P$ and $Q$ are said to be* homogeneous for $L$ with respect to $\mathfrak{F}$ *iff*

$$\big|\big(Act(P) \cup Act(Q)\big) \cap [a]\big| \leq 1, \qquad \textit{for all } a \in L.$$

Essentially, we require that both $P$ and $Q$ be able to perform at most one of the synchronisation actions belonging to the same element of $\mathfrak{F}$. For instance, let $P \triangleq a_1.\mathbf{0} + a_2.\mathbf{0}$, $Q \triangleq a_1.Q + a_2.Q$, and $\{a_1, a_2\} \in \mathfrak{F}$. Then $P$ and $Q$ are not homogeneous for $L = \{a_1, a_2\}$ in $\mathfrak{F}$. Let us now consider their respectively dimmed similar processes, $\bar{P} \triangleq a_1.\mathbf{0}$ and $\bar{Q} \triangleq a_1.\bar{Q}$. In this case, instead, it holds that $\bar{P}$ and $\bar{Q}$ are homogeneous for $L$ in $\mathfrak{F}$.

**Theorem 4 (Compositionality for Dimmed Simulation).** *Let $P$, $Q$ be two processes such that $P \preceq_{\mathfrak{F}} Q$. Then it holds that:*

*i)* $a.P \preceq_{\mathfrak{F}} b.Q$ *for all* $a, b$ *such that* $[a] = [b]$*;*
*ii)* $P + R \preceq_{\mathfrak{F}} Q + S$*, for any* $R$*,* $S$ *such that* $R \preceq_{\mathfrak{F}} S$*;*
*iii)* $P[g] \preceq_{\mathfrak{F}} Q[g]$ *for any* $g$ *pp-function for* $\mathfrak{F}$*;*
*iv)* $P/L \preceq_{\mathfrak{F}} Q/L$ *if* $L$ *is block coherent with* $\mathfrak{F}$*;*
*v)* $P \parallel_L R \preceq_{\mathfrak{F}} Q \parallel_L S$*, for* $R$*,* $S$ *such that* $R \preceq_{\mathfrak{F}} S$*, if* $L$ *is block coherent with* $\mathfrak{F}$ *and* $Q$ *and* $S$ *are homogeneous for* $L$ *with respect to* $\mathfrak{F}$*.*

The conditions required for the last point deserve more explanation. In general, dimmed simulation is not preserved if only $L$ is block coherent with $\mathfrak{F}$ but $Q$ and $S$ *are not* homogeneous for $L$ with $\mathfrak{F}$. To show this take, for instance, $P := a_1.\mathbf{0} + a_2.\mathbf{0}$, $R := P$, $Q := a_1.\mathbf{0}$, and $S := a_2.\mathbf{0}$, with $L = \{a_1, a_2\}$ and $L \in \mathfrak{F}$. Similarly, if the condition of homogeneity is satisfied but $L$ is not block coherent with $\mathfrak{F}$, dimmed simulation may not be preserved either. For instance, take $P := a_1.\mathbf{0}$, $R := P$, $Q := a_2.\mathbf{0}$, $S := Q$, with $L = \{a_1\}$ and $\{a_1, a_2\} \in \mathfrak{F}$. Finally, we wish to point out the fact that homogeneity is to be satisfied only by the simulating process $Q \parallel_L S$. This is why we have preferred a statement in the form of item *v)* instead of the weaker "$P \preceq_{\mathfrak{F}} Q \implies P \parallel_L R \preceq_{\mathfrak{F}} Q \parallel_L R$ for $Q$, $R$ homogeneous for $L$ in $\mathfrak{F}$ and $L$ block coherent with $\mathfrak{F}$." Stated in this way, homogeneity for $Q$ and $R$ would imply some form of homogeneity also in

the simulated process $P \parallel_L R$. Indeed, $R$ cannot enable two or more alternative synchronisation actions within the same part, thus significantly reducing the class of composite processes $P \parallel_L R$ that can be simulated. This is because, since $L$ must be block coherent with $\mathfrak{F}$, it could in principle contain two actions $a_1, a_2$ belonging to the same part. But $R$ can enable only one of them, say $a_1$. Therefore there would be $a_2 \in L$ which is never performed by one of the two operands. This, in turn, would cause $a_2$ never to be seen at all by $P \parallel_L R$.

Let us apply of dimmed simulation to our previous examples, showing that they can be simplified *compositionally* via $\preceq_{\mathfrak{F}}$ (but not with $\sim_{\mathfrak{F}}$).

*Example 2 (continued).* It possible to show that $P_2 \sim_{\mathfrak{F}} P_1$. Theorem 3 could be used to show that $P_1[m_1] \parallel_\emptyset P_2[m_2] \sim_{\mathfrak{F}} P_1[m_1 + m_2]$. Similarly, it holds that $B[k] \sim_{\mathfrak{F}} B_1[k]$, where $B_1 \triangleq put_1.get_1.B_1$. However, although $(P_1[m_1] \parallel_\emptyset P_2[m_2]) \parallel_{L_1} B[k] \sim_{\mathfrak{F}} P_1[m_1 + m_2] \parallel_{L_1} B_1[k]$ does hold, this fact *cannot* be inferred compositionally from Theorem 3 because $L_1$ is not singleton coherent with $\mathfrak{F}$. Thus, a simpler dimmed bisimilar process to $M$ cannot be obtained by congruence. Instead, a dimmed similar process can indeed be constructed compositionally. Since $\sim_{\mathfrak{F}}$ implies $\preceq_{\mathfrak{F}}$, we have that $P_1[m_1] \parallel_\emptyset P_2[m_2] \preceq_{\mathfrak{F}} P_1[m_1 + m_2]$ and $B[k] \preceq_{\mathfrak{F}} B_1[k]$. Now, $P_1[m_1 + m_2]$ and $B_1[k]$ are homogenous for $L_1$ and $L_1$ is block coherent with the chosen $\mathfrak{F}$. Hence, item *v)* of Theorem 4 yields

$$\big(P_1[m_1] \parallel_\emptyset P_2[m_2]\big) \parallel_{L_1} B[k] \preceq_{\mathfrak{F}} P_1[m_1 + m_2] \parallel_{L_1} B_1[k].$$

Analogously, it holds that $C_1[n_1] \parallel_\emptyset C_2[n_2] \preceq_{\mathfrak{F}} C_1[n_1 + n_2]$. Again, we have that $P_1[m_1 + m_2] \parallel_{L_1} B_1[k]$ and $C_1[n_1 + n_2]$ are homogeneous for $L_2$ and $L_2$ is block coherent with $\mathfrak{F}$. Therefore we are able to conclude (compositionally) that

$$M \preceq_{\mathfrak{F}} \bar{M}, \qquad \text{with} \quad \bar{M} := P_1[m_1 + m_2] \parallel_{L_1} B_1[k] \parallel_{L_2} C_1[n_1 + n_2].$$

*Example 3 (continued).* Similarly to the previous example, in Section 3 we discussed that

$$Sc \parallel_L P_1 \parallel_L P_2 \parallel_L P_3 \sim_{\mathfrak{F}} \bar{Sc} \parallel_L \bar{P} \parallel_L \bar{P} \parallel_L \bar{P}.$$

However, compositional reasoning was not possible because $L$ is not singleton coherent. Once again, $L$ is instead block coherent for the action partition $\mathfrak{F}$ chosen in this case. Now, it holds that $Sc \preceq_{\mathfrak{F}} \bar{Sc}$ and $P_i \preceq_{\mathfrak{F}} \bar{P}$. This implies that the relation $Sc \parallel_L P_1 \parallel_L P_2 \parallel_L P_3 \preceq_{\mathfrak{F}} \bar{Sc} \parallel_L \bar{P} \parallel_L \bar{P} \parallel_L \bar{P}$ can indeed be proven by repeatedly using point *v)* of Theorem 4.

## 5 Related Work

At the very core of our dimmed relations is an abstraction operated at the level of the transition system generated from a process term, which lifts concrete actions $a$, $b$, ..., to elements $[a]$, $[b]$, ..., of a given partition of the action set. As such, this work is in the general context of abstract interpretation [10], where a model approximation can preserve properties of interest, for example expressed as logical formulae (e.g., [11–13]). This framework has also been considered in process

algebra as early as in [14], where an approximation based on a preorder on the action set of CCS is studied. Although dimmed bisimilarity is a special case of the *extended bisimilarity* in [14], our approach is novel with respect to the use of such an abstraction. Indeed, [14] focuses on *partial specification*, i.e., replacing a process with another that can perform an *extended* action that can mimic any other concrete one. Partial specification is inherently non-symmetric since the identity of the concrete action is lost, differently from our dimmed relations where we preserve the information on its partition block. More recently, Galpin presented an analogous extension to bisimulation for process algebra for biochemical systems [15]. However, when used for state-space reduction purposes, her notion cannot be compared to our dimmed relations because [15] requires further conditions on the rate parameters in order to have congruence, and to obtain reduced models that can be related stochastically to the original ones.

The present paper is also related to [16], where the authors consider abstract interpretation to reduce infinite branching to finite branching of value-passing LOTOS based on trace semantics to enable model checking.

A large body of literature has also focused on action *refinement*, dual to the notion of action abstraction, where the main idea is that an atomic action is expanded into a process, e.g., a more detailed sequence of actions [17–19].

## 6 Conclusion

Dimmed behavioural relations permit trading-off between a detailed knowledge of the action types exhibited by a concrete model under study and a potentially more compact description arising from collapsing several actions together.

From a theoretical standpoint, the characterisation in terms of actions relabelling seems to do justice to this classic process algebra operator, which has been often neglected in recent developments of this field (see [20] for a discussion). The property of partition-preservation for a relabelling function presented in this paper is less restrictive than injectivity, as required for standard bisimulation results; yet it permits compositional reasoning for our dimmed relations.

From a more pragmatic standpoint, on a number of modelling patterns of practical interest, we showed that our dimmed relations can be effectively employed for a significantly more efficient (yet more abstract) analysis of heterogeneous systems, when heterogeneity is captured by the presence of analogous but formally distinct behaviours which are told apart by the use of distinct actions.

Future work will be concerned with a thorough investigation of a logical characterisation of dimmed bisimulation and simulation. We expect, however, that a straightforward adaptation of Hennessy-Milner logic should characterise the former; instead, an extension of simulation to *ready simulation* should be characterised by a suitably revised class of *denial formulas*, along the lines of [21].

***Acknowledgement.*** Luca Aceto, Jane Hillston, and Davide Sangiorgi are gratefully acknowledged for very insightful comments on an earlier draft of this paper. Most of this work was done while G.I. and M.T. were at LMU Munich and R.d.N.

was visiting; the authors would like to thank Martin Wirsing and his group for the excellent scientific and social atmosphere. This work was partially supported by the DFG project FEMPA and by the EU project QUANTICOL, 600708.

## References

1. Milner, R.: A Calculus of Communicating Systems. Springer-Verlag (1980)
2. Bošnački, D., Donaldson, A., Leuschel, M., Massart, T.: Efficient approximate verification of PROMELA models via symmetry markers. In: Automated Technology for Verification and Analysis. Volume 4762 of LNCS. Springer (2007) 300–315
3. Roscoe, A.: The Theory and Practice of Concurrency. Prentice Hall (1997)
4. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM **51**(1) (January 2008) 107–113
5. Aldini, A., Bernardo, M., Corradini, F.: A Process Algebraic Approach to Software Architecture Design. Springer Publishing Company (2009)
6. Paige, R., Tarjan, R.: Three partition refinement algorithms. SIAM Journal on Computing **16**(6) (1987) 973–989
7. Tribastone, M., Duguid, A., Gilmore, S.: The PEPA Eclipse Plug-in. Performance Evaluation Review **36**(4) (March 2009) 28–33
8. Milner, R.: An algebraic definition of simulation between programs. In: IJCAI. (1971) 481–489
9. Henzinger, M.R., Henzinger, T.A., Kopke, P.W.: Computing simulations on finite and infinite graphs. In: FOCS, IEEE Computer Society (1995) 453–462
10. Cousot, P., Cousot, R.: Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: POPL, New York, NY, USA, ACM (1977) 238–252
11. Clarke, E.M., Grumberg, O., Long, D.E.: Model checking and abstraction. ACM Trans. Program. Lang. Syst. **16**(5) (September 1994) 1512–1542
12. Dams, D., Gerth, R., Grumberg, O.: Abstract interpretation of reactive systems. ACM Trans. Program. Lang. Syst. **19**(2) (March 1997) 253–291
13. Fecher, H., Huth, M.: Model checking for action abstraction. In: Verification, Model Checking, and Abstract Interpretation. Volume 4905 of LNCS. Springer (2008) 112–126
14. Thomsen, B.: An extended bisimulation induced by a preorder on actions. M.Sc. thesis, Aalborg University (1987)
15. Galpin, V.: Equivalences for a biological process algebra. TCS **412**(43) (2011) 6058–6082
16. Fantechi, A., Gnesi, S., Latella, D.: Towards automatic temporal logic verification of value passing process algebra using abstract interpretation. In: CONCUR '96. Volume 1119 of LNCS., Springer (1996) 563–578
17. Glabbeek, R., Goltz, U.: Equivalence notions for concurrent systems and refinement of actions. In: MFCS. Volume 379 of LNCS. Springer (1989) 237–248
18. Aceto, L., Hennessy, M.: Towards action-refinement in process algebras. Information and Computation **103**(2) (1993) 204–269
19. Gorrieri, R., Rensink, A., Zamboni, M.A.: Action refinement. In: Handbook of Process Algebra, Elsevier (2000) 1047–1147
20. Sangiorgi, D.: Introduction to Bisimulation and Coinduction. Cambridge University Press (2011)
21. Bloom, B., Istrail, S., Meyer, A.R.: Bisimulation can't be traced. J. ACM **42**(1) (January 1995) 232–268