The Joys of Bisimulation

Colin Stirling

Department of Computer Science,
University of Edinburgh,
Edinburgh EH9 3JZ, UK,
email: cps@dcs.ed.ac.uk

1 Introduction

Bisimulation is a rich concept which appears in various areas of theoretical computer science. Its origins lie in concurrency theory, for instance see Milner [20], and in modal logic, see for example van Benthem [3].

In this paper we review results about bisimulation, from both the point of view of automata and from a logical point of view. We also consider how bisimulation has a role in finite model theory, and we offer a new undefinability result.

2 Basics

Labelled transition systems are commonly encountered in operational semantics of programs and systems. They are just labelled graphs. A transition system is a pair $\mathcal{T} = (\mathcal{S}, \{\stackrel{a}{\longrightarrow}:\ a \in \mathcal{A}\})$ where $\mathcal{S}$ is a non-empty set (of states), $\mathcal{A}$ is a non-empty set (of labels) and for each $a \in \mathcal{L}$, $\stackrel{a}{\longrightarrow}$ is a binary relation on $\mathcal{S}$. We write $s \stackrel{a}{\longrightarrow} s'$ instead of $(s, s') \in \stackrel{a}{\longrightarrow}$. Sometimes there is extra structure in a transition system, a set of atomic colours $\mathcal{Q}$, such that each colour $q \subseteq \mathcal{S}$ (the subset of states with colour q).

Bisimulations were introduced by Park [23] as a small refinement of the behavioural equivalence defined by Hennessy and Milner in [14] between basic CCS processes (whose behaviour is a transition system).

Definition 1 A binary relation $\mathcal{R}$ between states of a transition system is a *bisimulation* just in case whenever $(s, t) \in \mathcal{R}$ and $a \in \mathcal{A}$,

1. if $s \stackrel{a}{\longrightarrow} s'$ then $t \stackrel{a}{\longrightarrow} t'$ for some t' such that $(s', t') \in \mathcal{R}$ and
2. if $t \stackrel{a}{\longrightarrow} t'$ then $s \stackrel{a}{\longrightarrow} s'$ for some s' such that $(s', t') \in \mathcal{R}$.

In the case of an enriched transition system with colours there is an extra clause in the definition of a bisimulation that it preserves colours: if $(s, t) \in \mathcal{R}$ then

0. for all colours q, $s \in q$ iff $t \in q$

Simple examples of bisimulations are the identity relation and the empty relation. Two states of a transition system s and t are *bisimulation equivalent* (or *bisimilar*), written $s \sim t$, if there is a bisimulation relation $\mathcal{R}$ with $(s,t) \in \mathcal{R}$.

One can also present bisimulation equivalence as a game $\mathcal{G}(s_0,t_0)$, see for example [30, 28], which is played by two participants, players I and II. A *play* of $\mathcal{G}(s_0,t_0)$ is a finite or infinite length sequence of the form $(s_0,t_0)\ldots(s_i,t_i)\ldots$. Player I attempts to show that the initial states are different whereas player II wishes to establish that they are equivalent. Suppose an initial part of a play is $(s_0,t_0)\ldots(s_j,t_j)$. The next pair (s_{j+1},t_{j+1}) is determined by one of the following two moves:

- Player I chooses a transition $s_j \xrightarrow{a} s_{j+1}$ and then player II chooses a transition with the same label $t_j \xrightarrow{a} t_{j+1}$,
- Player I chooses a transition $t_j \xrightarrow{a} t_{j+1}$ and then player II chooses a transition with the same label $s_j \xrightarrow{a} s_{j+1}$.

The play continues with further moves. Player I always chooses first, and then player II, with full knowledge of player I's selection, must choose a corresponding transition of the other state.

A play of a game continues until one of the players wins. In a position (s,t) if one of these states has an a transition and the other doesnt then s and t are clearly distinguishable (and in the case of an enriched transition systems if one of these states has a colour which the other doesnt have then again they are distinguishable). Consequently any position (s_n,t_n) where s_n and t_n are distinguishable counts as a win for player I, and are called I-wins. A play is won by player I if the play reaches a I-win position. Any play that fails to reach such a position counts as a win for player II. Consequently player II wins if the play is infinite, or if the play reaches the position (s_n,t_n) and neither state has an available transition.

Different plays of a game can have different winners. Nevertheless for each game one of the players is able to win any play irrespective of what moves her opponent makes. To make this precise, the notion of strategy is essential. A strategy for a player is a family of rules which tell the player how to move. However it turns out that we only need to consider *history-free* strategies whose rules do not depend on what happened previously in the play. For player I a rule is therefore of the form "at position (s,t) choose transition x" where x is $s \xrightarrow{a} s'$ or $t \xrightarrow{a} t'$ for some a. A rule for player II is "at position (s,t) when player I has chosen x choose y" where x is either $s \xrightarrow{a} s'$ or $t \xrightarrow{a} t'$ and y is a corresponding transition of the other state. A player uses the strategy π in a play if all her moves obey the rules in π. The strategy π is a *winning strategy* if the player wins every play in which she uses π.

Proposition 1 *For any game $\mathcal{G}(s,t)$ either player I or player II has a history-free winning strategy.*

Proposition 2 *Player II has a winning strategy for $\mathcal{G}(s,t)$ iff $s \sim t$.*

Transition systems are models for basic process calculi, such as CCS and CSP. Models for richer calculi capturing value passing, mobility, causality, time, probability and locations have been developed. The basic notion of bisimulation has been generalised, often in a variety of different ways, to cover these extra features. Bisimulation also has a nice categorical representation via co-algebras due to Aczel, see for example [25], which allows a very general definition. It is an interesting question whether all the different brands of bisimulation are instances of this categorical account. In this paper we shall continue to examine only the very concrete notion of bisimulation on transition systems.

3 Bisimulation closure and invariance

It is common to identify a root of a transition system (as some special start state). Above we defined a bisimulation on states of the same transition graph. Equally we could have defined it between states of different transition systems. When transition systems are rooted we can then say that two systems are bisimilar if their roots are.

A family Δ of rooted transition graphs is said to be *closed under bisimulation equivalence* when the following holds:

$$\text{if } \mathcal{T} \in \Delta \text{ and } \mathcal{T} \sim \mathcal{T}' \text{ then } \mathcal{T}' \in \Delta$$

Given a rooted transition system there is a "smallest" transition system which is bisimilar to it: this is its *canonical* transition graph which is the result of first removing any states which are not reachable from the root, and then identifying bisimilar states (using quotienting).

An alternative perspective on bisimulation closure is from the viewpoint of properties of transition systems. Properties whose transition systems are bisimulation closed are said to be *bisimulation invariant.* Over rooted transition graphs, property Φ is bisimulation invariant provided that:

$$\text{if } \mathcal{T} \models \Phi \text{ and } \mathcal{T} \sim \mathcal{T}' \text{ then } \mathcal{T}' \models \Phi$$

(By $\mathcal{T} \models \Phi$ we mean that Φ is true of the transition graph $\mathcal{T}$.) On the whole, "counting" properties are not bisimulation invariant, for example "has 32 states" or "has an even number of states". In contrast temporal properties are bisimulation invariant, for instance "will eventually do an a-transition" or "is never able to do a b-transition". Other properties such as "has an Hamiltonian circuit" or "is 3-colourable" are also not bisimulation invariant. Later we shall be interested in parameterised properties, that is properties of arbitrary arity. We say that an n-ary property $\Phi(x_1, \ldots, x_n)$ on transition systems is bisimulation invariant provided that:

$$\begin{array}{l}\text{if } \mathcal{T} \models \Phi[s_1, \ldots, s_n] \text{ and } t_1, \ldots, t_n \text{ are states of } \mathcal{T}' \text{ and} \\ t_i \sim s_i \text{ for all } i : 1 \leq i \leq n \text{ then } \mathcal{T}' \models \Phi[t_1, \ldots, t_n]\end{array}$$

(By $\mathcal{T} \models \Phi[s_1, \ldots, s_n]$ we mean that Φ is true of the states $s_1, \ldots, s_n$ of $\mathcal{T}$). An example of a property which is not bisimulation invariant is "$x_1 \ldots x_n$ is a cycle", and an example of a bisimulation invariant property is "x_1 is language equivalent to x_2".

The notions of bismulation closure and invariance have appeared independently in a variety of contexts, see for instance [2, 3, 4, 7, 22].

4 Caucal's hierarchy

Bisimulation equivalence is a very fine equivalence between states. An interesting line of enquiry is to re-consider classical results in automata theory, replacing language equivalence with bismulation equivalence. These results concern definability, closure properties and decidability/undecidability.

Grammars can be viewed as generators of transition systems. Let Γ be a finite family of nonterminals and assume that $\mathcal{A}$ is a finite set (of terminals). A basic transition has the form $\alpha \xrightarrow{a} \beta$ where $\alpha, \beta \in \Gamma^*$ and $a \in \mathcal{A}$. A state is then any member of Γ^*, and the transition relations on states are defined as the least relations closed under basic transitions and the following prefix rule:

$$\text{PRE} \ \text{ if } \alpha \xrightarrow{a} \beta \text{ then } \alpha\delta \xrightarrow{a} \beta\delta$$

Given a state α we can define its rooted transition system whose states are just the ones reachable from α.

In the table below is a Caucal hierarchy of transition graph descriptions according to how the family of basic transitions is specified. In each case we assume a finite family of rules. Type 3 captures finite-state graphs, Type 2 captures context-free grammars in Greibach normal form, and Type $1\frac{1}{2}$, in fact, captures pushdown automata. For Type 0 and below this means that in each case there are finitely many basic transitions. In the other cases R_1 and R_2 are regular expressions over the alphabet Γ. The idea is that each rule $R_1 \xrightarrow{a} \beta$ stands for the possibly infinite family of basic transitions $\{\alpha \xrightarrow{a} \beta : \alpha \in R_1\}$ and $R_1 \xrightarrow{a} R_2$ stands for the family $\{\alpha \xrightarrow{a} \beta : \alpha \in R_1 \text{ and } \beta \in R_2\}$. For instance a Type -1 rule of the form $X^*Y \xrightarrow{a} Y$ includes for each $n \geq 0$ the basic transition $X^nY \xrightarrow{a} Y$.

	Basic Transitions
Type -2	$R_1 \xrightarrow{a} R_2$
Type -1	$R_1 \xrightarrow{a} \beta$
Type 0	$\alpha \xrightarrow{a} \beta$
Type $1\frac{1}{2}$	$\alpha \xrightarrow{a} \beta$ where $\lvert\alpha\rvert = 2$ and $\lvert\beta\rvert > 0$
Type 2	$X \xrightarrow{a} \beta$
Type 3	$X \xrightarrow{a} Y$ or $X \xrightarrow{a} \epsilon$

This hierarchy is implicit in Caucal's work on understanding context-free graphs, and understanding when the monadic second-order theory of graphs is decidable [5, 4, 6]. With respect to language equivalence, the hierarchy collapses to just two levels, the regular and the context free. The families between, and including, Type 2 and Type -2 are equivalent.

The standard transformation from pushdown automata to context free grammars (Type $1\frac{1}{2}$ to Type 2) does not preserve bisimulation equivalence. In fact, with respect to bisimilarity pushdown automata is a richer family than context free grammars. For instance, normed[1] Type 2 transition systems are closed under canonical transition systems. Caucal and Monfort [7] show that this is not true for Type $1\frac{1}{2}$ transition systems: see [4] for further results about canonical transition graphs. Caucal showed in [5] that Type 0 transition systems coincide (up to isomorphism) with Type $1\frac{1}{2}$. There is a strict hierarchy between Type 0 and Type -2. Therefore, with respect to bisimulation equivalence there are five levels in the hierarchy.

Baeten, Bergstra and Klop proved that bisimulation equivalence is decidable on normed Type 2 transition systems [1]. The decidability result was generalized in [9] to encompass all Type 2 graphs. Groote and Hüttel proved that other standard equivalences (traces, failures, simulation, 2/3-bisimulation etc..,) on Type 2 graphs are all undecidable [13]. The most recent result is by Sénizergues [27], who shows that bisimulation equivalence is decidable on Type -1 transition systems (which generalises his proof of decidability of language equivalence for DPDA [26]). This leaves as an open question whether it is also decidable for Type -2 systems.

One can build an alternative hierarchy when a sequence $\alpha \in \Gamma^*$ is viewed as a *multiset*. In which case the rule PRE above is to be understood as if $\alpha \xrightarrow{a} \beta$ then $\alpha \cup \delta \xrightarrow{a} \beta \cup \delta$ where $\cup$ is multiset union. Christensen, Hirshfeld and Moller showed that bisimulation equivalence is decidable on Type 2 graphs [8]. Hüttel proved that other equivalences are undecidable [16]. Type 0 graphs are Petri nets. Jančar showed undecidability of bisimilarity on Petri nets [17]. Under this commutative interpretation, Type 0 and Type $1\frac{1}{2}$ transition systems are not equivalent. Hirshfeld (utilizing Jančar's technique) showed undecidability of bisimulation for Type $1\frac{1}{2}$ systems, for more details see the survey [21].

5 Logics

Bisimulations were independently introduced in the context of modal logic by van Benthem [2]. A variety of logics can be defined over transition graphs.

Let M be the following family of modal formulas where a ranges over $\mathcal{A}$:

$$\Phi \ ::= \ \mathtt{tt} \ \mid \ \neg\,\Phi \ \mid \ \Phi_1 \vee \Phi_2 \ \mid \ \langle a \rangle \Phi$$

The inductive stipulation below defines when a state s has a modal property Φ, written $s \models_{\mathcal{T}} \Phi$, however we drop the index $\mathcal{T}$.

[1] A state t is terminal if it has no transitions. A state s is *normed* if for all s' such that $s \xrightarrow{w} s'$ for some $w \in A^*$, then there is a terminal t such that $s' \xrightarrow{u} t$ for some $u \in A^*$.

$$
\begin{array}{lll}
s \models \mathtt{tt} & & \\
s \models \neg \Phi & \text{iff} & s \not\models \Phi \\
s \models \Phi \vee \Psi & \text{iff} & s \models \Phi \text{ or } s \models \Psi \\
s \models \langle a \rangle \Phi & \text{iff} & \exists t.\, s \xrightarrow{a} t \text{ and } t \models \Phi
\end{array}
$$

This modal logic is known as Hennessy-Milner logic [14]. In the context of an enriched transition system one adds propositions q for each colour $q \in Q$ to the logic, with semantic clause: $s \models q$ iff $s \in q$.

Bisimilar states have the same modal properties. Let $s \equiv_M t$ just in case s and t have the same modal properties.

Proposition 1 *If* $s \sim t$ *then* $s \equiv_M t$.

The converse of Proposition 1 holds for a restricted set of transition systems. A state s is immediately image-finite if for each $a \in \mathcal{A}$ the set $\{t : s \xrightarrow{a} t\}$ is finite. And s is *image-finite* if every member of $\{t : \exists w \in \mathcal{A}^*.\, s \xrightarrow{w} t\}$ is immediately image-finite.

Proposition 2 *If* s *and* t *are image-finite and* $s \equiv_M t$ *then* $s \sim t$.

These two results are known as the modal *characterisation* of bisimulation equivalence, due to Hennessy and Milner [14]. (There is also an unrestricted characterisation result for infinitary modal logic. And there are less restrictive notions than image-finiteness for when characterisation holds, see [12, 15].)

The modal logic M is not very expressive. For instance it cannot define safety or liveness properties on transition systems which have been found to be very useful when analysing the behaviour of concurrent systems. Modal mu-calculus, μM, introduced by Kozen [19], has the rquired extra expressive power. The new constructs over and above those of M are:

$$\Phi \ ::= \ Z \ \mid \ \ldots \ \mid \ \mu Z.\, \Phi$$

where Z ranges over a family of propositional variables, and in the case of $\mu Z.\, \Phi$ there is a restiction that all free occurrences of Z in Φ are within the scope of an even number of negations (to guarantee monotonicity).

The semantics of M is extended to encompass the least fixed point operator μZ. Because of free variables valuations, $\mathcal{V}$, are used which assign to each variable Z a subset of states in $\mathcal{S}$. Let $\mathcal{V}[S/Z]$ be the valuation $\mathcal{V}'$ which agrees with $\mathcal{V}$ everywhere except Z when $\mathcal{V}'(Z) = S$. The inductive definition of satisfaction stipulates when a process E has the property Φ relative to $\mathcal{V}$, written $E \models_{\mathcal{V}} \Phi$, and the semantic clauses for the modal fragment are as before (except for the presence of $\mathcal{V}$).

$$
\begin{array}{ll}
s \models_{\mathcal{V}} Z & \text{iff } s \in \mathcal{V}(Z) \\
s \models_{\mathcal{V}} \mu Z.\, \Phi & \text{iff } \forall S \subseteq \mathcal{S}.\, \text{if } s \notin S \text{ then } \exists t \in \mathcal{S}.\, t \notin S \text{ and } t \models_{\mathcal{V}[S/Z]} \Phi
\end{array}
$$

The stipulation for the fixed point follows directly from the Tarski-Knaster theorem, as a least fixed point is the intersection of all prefixed points. (Again we would add atomic formulas q if we are interested in extended transition systems.)

The bisimulation characterisation result above, Propositions 1 and 2, remain true for closed formulas of μM.

Second-order propositional modal logic, 2M, is defined as an extension of M as follows:

$$\Phi \ ::= \ Z \ \mid \ \ldots \ \mid \ \Box\Phi \ \mid \ \forall Z.\Phi$$

The modality $\Box$ is the reflexive and transitive closure of $\bigcup\{[a] \,:\, a \in \mathcal{A}\}$, and is included so that 2M includes μM. As with modal mu-calculus we define when $s \models_{\mathcal{V}} \Phi$. The new clauses are:

$$\begin{array}{l} s \models_{\mathcal{V}} \Box\Phi \quad \text{iff } \forall t.\,\forall w \in \mathcal{A}^*.\text{ if } s \xrightarrow{w} t \text{ then } t \models_{\mathcal{V}} \Phi \\ s \models_{\mathcal{V}} \forall Z.\Phi \text{ iff } \forall S \subseteq \mathcal{S}.\ s \models_{\mathcal{V}[S/Z]} \Phi \end{array}$$

The operator $\forall Z$ is a set quantifier, ranging over subsets of $\mathcal{S}$. There is a straightforward translation of μM into 2M. Let Tr be this translation. The important case is the fixed point: $\mathrm{Tr}(\mu Z.\,\Phi) = \forall Z.(\Box(\mathrm{Tr}(\Phi) \to Z) \to Z)$.

Formulas of M and closed formulas of μM are bisimulation invariant (from Proposition 1 and its generalisation to μM). This is not true in the case of 2M, for it is too rich for characterising bisimulation: for instance, a variety of "counting" properties are definable, such as "has at least two different successors under an a transition" . This means that two bisimilar states need not have the same 2M properties.

Besides modal logics we can also consider other logics over transition systems. First-order logic, FOL, over transition systems contains binary relations E_a for each $a \in \mathcal{A}$ (and monadic predicates $q(x)$ for each colour q if extended transition systems are under consideration). Formulas have the form:

$$\Phi \ ::= \ xE_ay \ \mid \ x = y \ \mid \ \neg\,\Phi \ \mid \ \Phi_1 \wedge \Phi_2 \ \mid \ \forall x.\Phi$$

A formula $\Phi(x_1, \ldots, x_n)$ with at most free variables $x_1, \ldots, x_n$ will be true or false of transition system $\mathcal{T}$ and states $s_1, \ldots, s_n$ in the usual way.

Richer logics include first-order logic with fixed points, μFOL, where there is the extra formulas:

$$\Phi \ ::= \ Z(x_1, \ldots, x_k) \ \mid \ \ldots \ \mid \ \mu Z(x_1, \ldots, x_k).\,\Phi\,(y_1, \ldots, y_k)$$

In the case of $\mu Z(\ldots).\,\Phi\,(\ldots)$, there is the same restriction as in μM that all free occurrences of Z in Φ lie within the scope of an even number of negations.

An alternative extension of first-order logic is monadic second-order logic, 2OL, with the extra formulas:

$$\Phi \ ::= \ Z(x_1) \ \mid \ \ldots \ \mid \ \forall Z.\Phi$$

Van Benthem's use of bisimulation was to identify which formulas of FOL are equivalent to modal formulas (to M formulas), see the survey [3]. A formula $\Phi(x)$ is equivalent to a modal formula Φ' provided that for any $\mathcal{T}$ and for any state s, $\mathcal{T} \models \Phi[s]$ iff $s \models_{\mathcal{T}} \Phi'$.

Proposition 3 *A* FOL *formula $\Phi(x)$ over transition systems is bisimulation invariant iff Φ is equivalent to an* M *formula.*

This result was generalised by Janin and Walukiewicz [18] to 2OL and μM, as follows:

Proposition 4 *A* 2OL *formula $\Phi(x)$ over transition systems is bisimulation invariant iff it is equivalent to a closed μ*M *formula.*

One corollary of this result is that the bisimulation invariant (closed) formulas of 2M coincides with closed formulas of μM.

An interesting question is if there is also a characterisation of the bisimulation invariant formulas of μFOL. (See [22] for preliminary results but over finite models.)

6 Finite model theory

Finite model theory is concerned with relationships between complexity classes and logics over finite structures. It is interesting to consider bisimulation invariance in the context of finite model theory.

Rosen showed that Proposition 3 of the previous section remains true with the restriction to finite transition systems [24]. It is an open question whether Proposition 4 also remains true under this restriction.

Part of the interest in relationships between μM and 2M or 2OL with respect to finite transition systems is that within 2M and 2OL one can define NP-complete problems: examples include 3-colourability on finite connected undirected graphs. Consider such a graph. If there is an edge between two states s and t let $s \xrightarrow{a} t$ and $t \xrightarrow{a} s$. So in this case $\mathcal{A} = \{a\}$, and 3-colourability is given by:

$$\exists X. \exists Y. \exists Z. (\Phi \wedge \Box((X \rightarrow [a]\neg X) \wedge (Y \rightarrow [a]\neg Y) \wedge (Z \rightarrow [a]\neg Z)))$$

where Φ, which says that every vertex has a unique colour, is

$$\Box((X \wedge \neg Y \wedge \neg Z) \vee (Y \wedge \neg Z \wedge \neg X) \vee (Z \wedge \neg X \wedge \neg Y))$$

In contrast, μM formulas over finite transition systems can only express PTIME properties.

An interesting open question is whether there is a logic which captures exactly the PTIME properties of transition systems. Otto has shown that there is a logic for the PTIME properties that are bisimulation invariant [22]. The right setting is μFOL over canonical transition systems (where $=$ is $\sim$, and a linear ordering on states is thereby definable).

We now consider emaciated finite transition systems whose set $\mathcal{A}$ is a singleton. That is now $\mathcal{T} = (\mathcal{S}, \longrightarrow)$ where $\mathcal{S}$ is finite. We can define language equivalence on emaciated transition systems. Let $s \xrightarrow{n} t$, when $n \geq 0$, if there is a sequence of transitions of length n from s to t (and by convention $s \xrightarrow{0} s$). A state is terminal if it has no transitions. The language of state s is the set $L(s) = \{i \geq 0 : s \xrightarrow{i} t$ and t is terminal$\}$. Consequently, s and s' are language equivalent if $L(s) = L(s')$. The property "x is language equivalent to y" as was noted earlier is bisimulation invariant. Notice that this is an example of a dyadic invariant property.

Proposition 1 *Language equivalence on (canonical) finite transition graphs is co-NP complete.*

Hence language equivalence over finite transition systems is definable in μFOL iff PTIME = NP. Dawar offers a different route to this observation [10].

A classical result (due to Immermann, Gurevich and Shelah) in a slightly normalised form is:

Proposition 2 *A* μFOL *formula* $\Psi(y_1, \ldots, y_n)$ *over finite transition systems is equivalent to a formula of the form* $\exists u. (\mu Z(x_1, \ldots, x_m). \Phi(y_1, \ldots, y_n, \widetilde{u}))$ *where* Φ *is first-order and contains at most* $x_1, \ldots, x_m$ *free.*

The argument places in the application $(\ldots)$ from $n+1$ to m are all filled by the same element u. This allows for the arity of the defining fixed point m to be larger than the arity of the μFOL formula n.

Consequently, if one can prove that "y is language equivalent to z", $\Psi(y, z)$, is not definable by a μFOL formula in normal form, $\exists u. (\mu Z(x_1, \ldots, x_m). \Phi(y, z, \widetilde{u}))$, then this would show that PTIME is different from NP. As a first step, we have proved the following using tableaux:

Theorem 1 *Language equivalence* $\Psi(y, z)$ *is not definable in* μFOL *by a normal formula of the form* $\exists u. (\mu Z(x_1, x_2, x_3). \Phi(y, z, u))$.

Acknowledgement: I would like to thank Julian Bradfield and Anuj Dawar for help in understanding finite model theory.

References

1. Baeten, J., Bergstra, J., and Klop, J. (1993). Decidability of bisimulation equivalence for processes generating context-free languages. *Journal of Association of Computing Machinery*, **40**, 653-682.
2. van Benthem, J. (1984). Correspondence theory. In *Handbook of Philosophical Logic*, Vol. II, ed. Gabbay, D. and Guenthner, F., 167-248, Reidel.
3. van Benthem, J. (1996). Exploring Logical Dynamics. *CSLI Publications*.
4. Burkart, O., Caucal, D., and Steffen, B. (1996). Bisimulation collapse and the process taxonomy. *Lecture Notes in Computer Science*, **1119**, 247-262.
5. Caucal, D. (1992). On the regular structure of prefix rewriting. *Theoretical Computer Science*, **106**, 61-86.
6. Caucal, D. (1996). On infinite transition graphs having a decidable monadic theory. *Lecture Notes in Computer Science*, **1099**, 194-205.
7. Caucal, D., and Monfort, R. (1990). On the transition graphs of automata and grammars. *Lecture Notes in Computer Science*, **484**, 311-337.
8. Christensen, S., Hirshfeld, Y., and Moller, F. (1993). Bisimulation is decidable for basic parallel processes. *Lecture Notes in Computer Science*, **715**, 143-157.

9. Christensen, S., Hüttel, H., and Stirling, C. (1995). Bisimulation equivalence is decidable for all context-free processes. *Information and Computation*, **121**, 143-148.
10. Dawar, A. (1997). A restricted second-order logic for finite structures, To appear in *Information and Computation.*
11. Emerson, E., and Jutla, C. (1988). The complexity of tree automata and logics of programs. *Extended version from FOCS '88.*
12. Goldblatt, R. (1995). Saturation and the Hennessy-Milner property. In *Modal Logic and Process Algebra*, ed. Ponse, A., De Rijke, M. and Venema, Y. *CSLI Publications*, 107-130.
13. Groote, J., and Hüttel, H. (1994). Undecidable equivalences for basic process algebra. *Information and Computation*, **115**, 354-371.
14. Hennessy, M. and Milner, R. (1985). Algebraic laws for nondeterminism and concurrency. *Journal of Association of Computer Machinery*, **32**, 137-162.
15. Hollenberg, M. (1995). Hennessy-Milner classes and process calculi. In *Modal Logic and Process Algebra*, ed. Ponse, A., De Rijke, M. and Venema, Y. *CSLI Publications*, 187-216.
16. Hüttel, H. (1994). Undecidable equivalences for basic parallel processes. *Lecture Notes in Computer Science*, **789**.
17. Jančar, P. (1994). Decidability questions for bisimilarity of Petri nets and some related problems. *Lecture Notes in Computer Science*, **775**, 581-594.
18. Janin, D. and Walukiewicz, I (1996). On the expressive completeness of the propositional mu-calculus with respect to the monadic second order logic. *Lecture Notes in Computer Science*, **1119**, 263-277.
19. Kozen, D. (1983). Results on the propositional mu-calculus. *Theoretical Computer Science*, **27**, 333-354.
20. Milner, R. (1989). *Communication and Concurrency.* Prentice Hall.
21. Moller, F. (1996). Infinite results. *Lecture Notes in Computer Science*, **1119**, 195-216.
22. Otto, M. (1997). Bisimulation-invariant ptime and higher-dimensional μ-calculus. *Preliminary report RWTH Aachen.*
23. Park, D. (1981). Concurrency and automata on infinite sequences. *Lecture Notes in Computer Science*, **154**, 561-572.
24. Rosen, E. (1995). Modal logic over finite structures. *Tech Report*, University of Amsterdam.
25. Rutten, J. (1995). A calculus of transition systems (towards universal coalgebra). In *Modal Logic and Process Algebra*, ed. Ponse, A., De Rijke, M. and Venema, Y. *CSLI Publications*, 187-216.
26. Sénizergues, G. (1997). The equivalence problem for deterministic pushdown automata is decidable. *Lecture Notes in Computer Science*, **1256**, 671-681.
27. Sénizergues, G. (1998). $\Gamma(A) \sim \Gamma(B)$? Draft paper.
28. Stirling, C. (1996). Modal and temporal logics for processes. *Lecture Notes in Computer Science*, **1043**, 149-237.
29. Stirling, C. (1996). Games and modal mu-calculus. *Lecture Notes in Computer Science*, **1055**, 298-312.
30. Thomas, W. (1993). On the Ehrenfeucht-Fraïssé game in theoretical computer science. *Lecture Notes in Computer Science*, **668**.