entropy

MDPI

Article

Integrated Information in Process-Algebraic Compositions

Tommaso Bolognesi

Institute of Information Science and Technologies, National Research Council (ISTI-CNR), 1, Via Moruzzi, 56124 Pisa, Italy; t.bolognesi@isti.cnr.it

Received: 25 March 2019; Accepted: 14 August 2019; Published: 17 August 2019

Abstract: Integrated Information Theory (IIT) is most typically applied to *Boolean Nets*, a state transition model in which system parts cooperate by *sharing state variables*. By contrast, in *Process Algebra*, whose semantics can also be formulated in terms of (labeled) state transitions, system parts—"processes"—cooperate by *sharing transitions* with matching labels, according to interaction patterns expressed by suitable composition operators. Despite this substantial difference, questioning how much additional information is provided by the integration of the interacting partners above and beyond the sum of their independent contributions appears perfectly legitimate with both types of cooperation. In fact, we collect statistical data about ϕ—integrated information—relative to pairs of boolean nets that cooperate by *three* alternative mechanisms: shared variables—the standard choice for boolean nets—and *two forms* of shared transition, inspired by two process algebras. We name these mechanisms α, β and γ. Quantitative characterizations of all of them are obtained by considering three alternative *execution modes*, namely synchronous, asynchronous and "hybrid", by exploring the full range of possible coupling degrees in all three cases, and by considering two possible definitions of ϕ based on two alternative notions of distribution distance.

Keywords: boolean net; process algebra; parallel composition operator; integrated information

1. Introduction

Integrated Information Theory (IIT) [1,2] is concerned with the study of natural or artificial systems formed by many interconnected micro-components. One of the key steps in this study is the identification of the "hidden" macro-components of the system, namely its *Minimum Information Partition* (*MIP*). The macro-components of the *MIP* can be seen as distinct but interacting parts: ϕ measures the added value provided by their integration/composition with respect to the plain sum of their contributions—how much the integrated whole is more than the sum of the separate parts.

In *Boolean Nets* [3]—the state transition model predominantly used for illustrating IIT—the integration/cooperation among the *MIP* parts occurs via the directed edges that interconnect them: the parts influence one another by reading each other's boolean variables—a form of *shared-variable cooperation*.

In this paper, we contrast the above cooperation mechanism with an alternative one based on *shared transitions*, that arises in *Process Algebras* (or "*Calculi*") [4–8]. Furthermore, viewing the interacting partners under the process algebraic perspective has suggested us to extend our analysis to *three* execution modes for boolean nets: *synchronous*, *asynchronous* and "*hybrid*" (although the second one is soon dropped).

We have three main objectives in mind.

The first is to put the interaction mechanisms of *shared variables* and *shared transitions* on an equal footing, and to obtain some numerical characterization of their "performance" with respect to the ability to produce integrated information.

The second is related to the central application area of IIT—the modeling and quantification of the emergence of consciousness from the complex structure of the brain. Given that the brain architecture is indeed *intrinsically* structured into macro-components, the investigation of alternative or additional mechanisms of cooperation among them, that *explicitly* reflect such higher-level structure, could be an interesting complement to the study of cooperation mechanism that only address micro-components.

The third objective is relevant to the areas from which these additional cooperation mechanisms are borrowed, namely Process Algebra and, more generally, formal methods for Software Engineering. Using informational measures from IIT appears as a completely novel and attractive approach to characterizing quantitatively these practically useful mechanisms and their associated operators.

The paper is organized as follows.

In Section 2, we briefly recall the definition of Boolean Net and introduce the three execution modes: *synchronous*, *asynchronous* and *hybrid*. The first, yielding deterministic behaviors, is the traditional mode; however, the nondeterministic asynchronous and hybrid modes appear more in line with the nondeterminism of the systems typically addressed by Process Algebra. Here, we also discuss the three modes with respect to the property of *conditional independence*.

In Section 3, we introduce the interaction mechanism adopted in process-algebraic calculi/languages, one based on shared labeled transitions (abbreviated "*sharTrans*") as opposed to shared variables ("*sharVar*"). We in particular illustrate the flexible parametric operator of *parallel composition* from the LOTOS language, denoted "$|\beta|$": expression $P|\beta|Q$ describes a system composed of two *processes* P and Q that cooperate by sharing some transitions, where β defines the degree of coupling between them.

In Section 4, we show that the parallel composition operator $|\beta|$ can be readily used also for composing two *boolean nets* P and Q—still written $P|\beta|Q$—provided these are enriched with transition labels, and regardless of the chosen execution mode. This enables us to put the newly considered form of composition/integration under the lens of IIT without need to import and discuss any other element of Process Algebras.

In Section 5, we introduce notation $P{<}\alpha{>}Q$ and the idea to control the degree of coupling between two bool nets P and Q, under the *sharVar* mechanism, by controlling the number α of edges crossing between them.

In Section 6, we recall the notion of *integrated information*, the central concept of IIT, both in its *state-dependent form* $\phi(X)$ and in its *averaged form*, which we denote $\bar{\phi}$. These definitions are based on a distance function $d(y_{coop}, y_{indep})$ between two probabilistic state distributions, where y_{coop} reflects inter-part cooperation while y_{indep} corresponds to their independent operation. In IIT 2.0 [1], d is *Relative Entropy* (or *Kullback–Liebler divergence*, denoted *dkl*). We show that the definition of y_{indep} for the *sharVar* context is such to avoid the "*dkl-mismatch problem*" that may arise when applying *dkl* to generic distributions. Then, we conduct a statistical analysis of $\bar{\phi}^{mode}(P{<}\alpha{>}Q)$ in order to study its dependency on α for the *sync* and *hybrid* execution modes of $P{<}\alpha{>}Q$, using 10 pairs (P, Q) of randomly generated bool nets. For facilitating the comparison of $P{<}\alpha{>}Q$ with *sharTrans* compositions (in view of potential *dkl*-mismatch problems in the latter), we extend our statistical analysis by using a version of $\bar{\phi}$ in which *dkl* is replaced by *Manhattan distance*.

In Section 7, we address the problem of defining $\bar{\phi}$ in the very different context of *sharTrans* bool net compositions $P|\beta|Q$. Here, we have to face two problems: the presence of deadlocks and the mentioned *dkl*-mismatches. The first problem is solved easily; a drastic way to bypass the second one is to switch to the $\bar{\phi}$ variant based on Manhattan distance.

Wishing to stick to the original, *dkl*-based definition of $\bar{\phi}$, in Section 8, we consider an alternative, process-algebraic cooperation mechanism, borrowed from CCS (Calculus of Communicating Systems) [5], that avoids the *dkl*-mismatch problem. In fact, we combine CCS parallelism ("$P|Q$")

and restriction ("$\backslash\gamma$") into the convenient syntactic form $P[\gamma]Q \equiv (P|Q)\backslash\gamma$, where parameter γ still expresses the degree of coupling between the interacting parties. This enables us to compare, by statistical experiments, the trends of $\bar{\phi}$, in its original *dkl*-based definition [1], for $P{<}\alpha{>}Q$ and $P[\gamma]Q$.

In Section 9, we regroup the 15 plots introduced in the previous sections into a compact table that facilitates the comparison of mechanisms α, β and γ.

Some closing remarks are given in Section 10.

2. Boolean Nets: Sync, Async and Hybrid Execution Modes

Boolean nets [3] are discrete sequential dynamical systems. An (n, k)-*boolean net* ("bool net" in the sequel) is a pair $(G(B, E), F)$ where:

- $G(B, E)$ is a directed graph with n vertices $B = \{b_1, \ldots, b_n\}$, and edge set E; each vertex $b_i \in B$ has exactly k incoming edges (this limitation on node in-degree is not essential; we adopt it only for convenience of implementation and notation): $b_{i,1} \to b_i, \ldots, b_{i,k} \to b_i$, so that $|E| = nk$.
- $F = \{f_1, \ldots f_n\}$ is a set of n boolean functions of k arguments, one for each vertex in B.

Each vertex $b_i \in B$ is a boolean variable controlled by boolean function $f_i(b_{i,1} \ldots b_{i,k})$ from F, where the ordered k-tuple of arguments $(b_{i,1} \ldots b_{i,k})$ corresponds to the edges incident to b_i. In the sequel, an (n, k)-bool net P is sometimes denoted $P(n, k)$.

A bool net *computation* is a sequence of *steps*, assumed to take place in *discrete time*—one step at each clock tick. Each step consists of the instantaneous and simultaneous firing of a group of nodes, called the *firing group*. A firing group is a subset of B, which can be conveniently identified also by its characteristic function (i.e., characteristic function {1,1,0} indicates that only the first two nodes fire, out of three). When node b_i fires, its value is updated according to boolean function f_i.

Notation. Lower case letters x and y denote discrete *random variables*. In particular, x or $x(t)$ is the current state at time t of an (n, k)-bool net, consisting of an n-tuple of binary random variables $(b_1(t) \ldots b_n(t))$. Similarly, y or $y(t + 1) = (b_1(t + 1) \ldots b_n(t + 1))$ is the next state at time $(t + 1)$. Upper case letters X and Y denote actual n-tuples of bits, i.e., the values that variables x and y may assume: $X = (\gamma_1 \ldots \gamma_n)$ and $Y = (\delta_1 \ldots \delta_n)$, where γs and δs are bits. Subscript i in X_i is used when we want X_i to range in a set of n-tuples, for example in the whole set $\{0, 1\}^n$—*not for selecting an element inside the tuple!* For example, writing $prob(x = X_i)$, where $X_i = (\gamma_1^i \ldots \gamma_n^i)$, means $prob(b_1(t) = \gamma_1^i \ldots b_n(t) = \gamma_n^i)$. We consistently use identifiers x and X for predecessor states, and y and Y for successor states.

The *densities* of random variables x and y, often called here "*distributions*", are denoted p_x and p_y, but sometimes also x and y, with symbol overloading; the meaning should be clear from the context. For example, the probability for variable y to assume value Y_i is written $p_y(Y_i)$ but also $y(Y_i)$.

tpm. In the sequel, an essential role is played by the *transition probability matrix* (*tpm*), in which entry $tpm(X_{PQ}, Y_{PQ})$ expresses the conditional probability $prob(Y_{PQ}|X_{PQ})$ obtained by counting *all* possible transitions that lead from state X_{PQ} to state Y_{PQ}.

Given an (n, k)-bool net, we consider three *execution modes* for it, which differ in the way we define FG, the *set of firing groups* possible at each step. (Note that FG does not depend on the current state.)

Sync. All nodes fire (update) simultaneously. In other words, FG consists of only one firing group, which includes all n nodes. For example, when $n = 3$, we have $FG = \{B\}$ (using node sets), also represented as $FG = \{\{1, 1, 1\}\}$ (using characteristic functions). Evolution is *deterministic*: each global state has only one successor. *Sync* boolean nets are a generalization of Cellular Automata.

Async. Nodes fire one at a time, the choice being made by a uniform random distribution. In other words, FG consists of n firing groups, each being a singleton. For $n = 3$, we have $FG = \{\{b_1\}, \{b_2\}, \{b_3\}\}$, or FG = {{1,0,0}, {0,1,0}, {0,0,1}}. Evolution is *nondeterministic*: each global state may have multiple successors—as many as n (note that, with fixed current state, the correspondence between firing groups and next states can be many-to-one).

Hybrid. (I am thankful to Larissa Albantakis for having drawn my attention to this execution mode and its conditional independence property.) Here, FG consists of 2^n firing groups—namely, all the subsets of the node set B. For $n = 3$, using characteristic functions, we have FG = {{0,0,0}, {0,0,1}, {0,1,0} . . . {1,1,1}}. The choice is made, again, by a uniform random distribution: the probability to pick any specific firing group is $1/2^n$, where n is the number of nodes. Note that this is equivalent to firing each node with probability $1/2$, independently node by node. Evolution is *nondeterministic*: each global state may have multiple distinct successors—as many as 2^n. Note that the empty firing group is also included.

We write tpm^m_S for denoting the transition probability matrix of bool net S executed in mode m (*sync*, *async* or *hybrid*).

We soon deal with composite bool nets. The easiest way to compose two independent bool nets P and Q is to take their union, defined in the obvious way and denoted $P \cup Q$. P and Q are disconnected, and do not communicate. It is trivial to see that $P \cup Q$ is itself a bool net, which can be executed in any of the three modes.

Let us then establish some simple facts about the relations between the set $FG^m_{P\cup Q}$ of firing groups of $P \cup Q$ and sets FG^m_P and FG^m_Q of firing groups of the components, in the three modes. In the equations below, firing groups are conceived as node sets.

$$FG^{sync}_{P\cup Q} = \{B_P \cup B_Q\} = FG^{sync}_P \times_\cup FG^{sync}_Q \tag{1}$$

$$FG^{async}_{P\cup Q} = \{\{b_1\} \dots \{b_p\}\} \cup \{\{b'_1\} \dots \{b'_q\}\} = FG^{async}_P \cup FG^{async}_Q \tag{2}$$

$$FG^{hybrid}_{P\cup Q} = 2^{B_P \cup B_Q} = 2^{B_P} \times_\cup 2^{B_Q} = FG^{hybrid}_P \times_\cup FG^{hybrid}_Q. \tag{3}$$

In Equation (1), $\{B_P \cup B_Q\}$ is a *singleton* set—a set whose unique element is the set $B_P \cup B_Q$ of nodes. In *sync* mode, FG—be it referred to P, Q or $P \cup Q$—has only one element, namely the firing group involving all available nodes. Thus, $FG^{sync}_P = \{B_P\}$ and $FG^{sync}_Q = \{B_Q\}$. Symbol "$\times_\cup$" denotes a Cartesian product that takes the union of the paired elements, which are node sets (e.g., $\{A, B\} \times_\cup \{C, D\} = \{A \cup C, A \cup D, \dots\}$.

Executing $P \cup Q$ in *async* mode (Equation (2)) means to fire (update) one node at a time. Thus, in this equation, we make use of singleton sets (e.g., $\{b_i\}$ or $\{b'_j\}$) formed from the individual nodes of P and Q, where $B_P = \{b_1 \dots b_p\}$ and $B_Q = \{b'_1 \dots b'_q\}$. Set $FG^{async}_{P\cup Q}$ is then the plain union of sets FG^{async}_P and FG^{async}_Q.

Executing $P \cup Q$ in *hybrid* mode (Equation (3)) means to fire any possible subset of $B_P \cup B_Q$, including the empty set. This set of firing groups can also be seen as the "special" Cartesian product $FG^{hybrid}_P \times_\cup FG^{hybrid}_Q$.

Note that the FG of the whole system is a Cartesian product only for the *sync* and *hybrid* modes, and that these results clearly hold also when P and Q are connected by some edges, i.e., are not independent, since firing groups are defined relative to node sets, regardless of node interconnections.

Conditional Independence in the Three Modes

Let $x = \{b_1(t) \dots b_n(t)\}$ denote the current global state of an (n, k) bool net at time t, and $y = \{b_1(t + 1) \dots b_n(t + 1)\}$ be the next global state, at time $t + 1$.

Following Pearl [9], we say that, for any $i, j \in \{1 \dots n\}$, $b_i(t+1)$ is *conditionally independent from* $b_j(t+1)$, *given* x, if

$$p(b_i(t+1)|x, b_j(t+1)) = p(b_i(t+1)|x), \text{when } p(x, b_j(t+1)) > 0. \tag{4}$$

Once x is known, the additional knowledge of $b_j(t+1)$ does not add anything to what we already know about $b_i(t+1)$ (and vice versa). Note that the above equation means:

$$p(b_i(t+1) = \delta_i | x = (\gamma_1 \dots \gamma_n), b_j(t+1) = \delta_j) = p(b_i(t+1) = \delta_i | x = (\gamma_1 \dots \gamma_n)) \tag{5}$$

for all γs and δs such that $p(x = (\gamma_1 \dots \gamma_n), b_j(t+1) = \delta_j) > 0$.

Two random variables y_1 and y_2 are *independent* if and only if their *mutual information* [10] is null: $M(y_1, y_2) = 0$. Similarly, two random variables y_1 and y_2 are *conditionally independent*, given x, a third variable, if and only if their *conditional mutual information* is null: $M(y_1, y_2|x) = 0$.

Recall that *mutual information* $M(y_1, y_2)$, a symmetric quantity representing the information provided on average by one variable about the other, is:

$$M(y_1, y_2) = \sum_{i,j} p_{y_1 y_2}(Y_i, Y_j) Log_2 \frac{p_{y_1 y_2}(Y_i, Y_j)}{p_{y_1}(Y_i) p_{y_2}(Y_j)}, \tag{6}$$

where $p_{y_1 y_2}$ is the joint distribution of the two variables, while p_{y_1} and p_{y_2} are the respective marginal distributions.

The *conditional mutual information* between variables y_1 and y_2, *relative to variable* x, is

$$M(y_1, y_2|x) = \sum_{i,j,k} p_{y_1 y_2 x}(Y_i, Y_j, X_k) Log_2 \frac{p_{y_1 y_2 x}(Y_i, Y_j, X_k) p_x(X_k)}{p_{y_1 x}(Y_i, X_k) p_{y_2 x}(Y_j, X_k)}, \tag{7}$$

which can be also formulated as the weighted sum of the mutual information relative to the individual values X_k of variable x.

IIT attributes much importance to conditional independence: when the property is satisfied, each element b_i, with its function $f_i(b_{i,1} \dots b_{i,k})$, can be interpreted as an *individual causal element* within the system; when it is violated, a possibly undesirable form of instantaneous causal influence between $b_i(t+1)$ and $b_j(t+1)$ arises.

The three considered execution modes perform differently with respect to conditional independence.

- The *sync* mode entails conditional independence for the simple reason that, due to transition determinism, knowledge of the current state X already provides *complete* information about $b_i(t+1)$ (and $b_j(t+1)$).
- With the *async* mode, conditional independence is violated: knowing $b_j(t+1)$, in the case $b_j(t+1) \neq b_j(t)$, reveals that b_j has been the only firing (updating) node, which implies $b_i(t+1) = b_i(t)$—a conclusion that we cannot draw from the pure knowledge of x.
- The *hybrid* mode entails conditional independence. As already observed, picking a firing group with uniform probability $1/2^n$ is equivalent to firing each node with probability $1/2$, independently node by node. Thus, finding that b_j has fired does not provide additional information on whether or not b_i has fired, thus on $b_i(t+1)$.

It is straightforward to see that the above definition of conditional independence, and the results for the three modes, are valid not only for individual nodes but also for groups of nodes, i.e., for parts of the net, such as P and Q in the sequel.

Two-Step Conditional Independence

It could be of some interest to see how conditional dependence/independence carries over to the case of *two or more transitions*, e.g., for analyzing behaviors under macro-transitions, or temporal coarse-graining. Somewhat surprisingly, the scenario changes as follows.

Let $x(t)$, $y(t+1)$, $z(t+2)$ be a sequence of global states; we are now to compare $prob(b_i(t+2)|x(t), b_j(t+2))$ with $prob(b_i(t+2)|x(t))$.

In *sync* mode, conditional independence is still valid, for the same argument of the case of one transition: $z(t+2)$ is completely defined, once $x(t)$ is known.

In *async* mode, conditional independence is still violated. If $b_j(t+2) \neq b_j(t)$, we know that node j has fired at least once: this fact reduces the probability that node i has fired at time $t+1$ or $t+2$, providing us additional information about $b_i(t+2)$.

The change occurs with respect to the *hybrid* mode: while the property is satisfied after one transition, it is violated after two. Informally, finding $b_j(t+2) \neq b_j(t)$ reveals that node j has fired at least once, which yields additional information about $b_j(t+1)$. This, in turn, may provide additional information about $b_i(t+2)$ beyond what is already given by $x(t)$. Of course, knowing *who fired* in the first step still does not say anything about *who fired* in the second step: the point is that additional knowledge about the intermediate *values* of $y(t+1)$ does refine our knowledge about the possible final *values* of $z(t+2)$.

3. Parallel Composition of LOTOS Processes: $P|\beta|Q$

In Process Algebras [4–8], a distributed concurrent system is formally described as a set of interacting processes. Each of these formalisms offers its own set of *operators* for specifying actions, interactions, concurrency, choice, nondeterminism, recursion, etc. By the *Structural Operational Semantics* [11], the syntactic expressions built by these operators, describing system structure and behavior, can be formally interpreted as *labeled transition systems*.

Of crucial importance for specifying the macro-structure and interaction patterns of the system are the *parallel composition operators*. We in particular refer to the flexible, parametric parallel composition operator of the process-algebraic language *LOTOS* (Language of Temporal Ordering Specification) [8].

When two processes P and Q are composed by the parallel composition operator "$|\beta|$", where β is the set of "synchronization labels", the resulting labeled transition system is obtained by forcing the processes to proceed jointly—in synchrony—with the transitions with labels in β, while proceeding independently—in "interleaving"—with their other transitions.

The Structural Operational Semantics provides one or more axioms or inference rules specifying the transitions associated with (the expressions formed by) each operator. The inference rules are usually written as "fractions", and define the transitions of an expression formed by that operator, appearing in the "denominator" (the *conclusion*), in terms of the transitions of the operator arguments, appearing in the "numerator" (the *premise*).

Three inference rules define the semantics of the LOTOS parallel composition expression $P|\beta|Q$, where P and Q are themselves expressions (processes):

$$\frac{P \xrightarrow{x} P' \wedge x \notin \beta}{P|\beta|Q \xrightarrow{x} P'|\beta|Q} \quad (\textit{LOTOS left interleaving}) \tag{8}$$

$$\frac{Q \xrightarrow{x} Q' \wedge x \notin \beta}{P|\beta|Q \xrightarrow{x} P|\beta|Q'} \quad (\textit{LOTOS right interleaving}) \tag{9}$$

$$\frac{P \xrightarrow{x} P' \wedge Q \xrightarrow{x} Q' \wedge x \in \beta}{P|\beta|Q \xrightarrow{x} P'|\beta|Q'} \quad (\textit{LOTOS synchronization}) \tag{10}$$

For example, when two processes $P[a,b,c]$ and $Q[b,c,d]$, able to perform transitions with labels in, respectively, sets $\{a,b,c\}$ and $\{b,c,d\}$, are composed by the expression "$P[a,b,c]|\{b,c\}|Q[b,c,d]$", they will interleave their local transitions labeled a and d, and synchronize those labeled b and c.

When the set of synchronization labels is empty—$\beta = \emptyset$—we have the special case of *pure interleaving* composition $P|\emptyset|Q$, also denoted $P|||Q$, where "$|||$" is called the *interleaving* operator. In this case, it is clear that the rules in Equations (8) and (9) are still applicable while the rule in Equation (10) is not; thus, in composition $P|||Q$, the components can only proceed one at a time.

In the next section, we discuss how to apply the above parallel composition operator to bool nets, and the way this operator performs with respect to the conditional independence property.

4. Parallel Compositions of Bool Nets: $P|\beta|Q$

Bool nets are state transition systems, and since the rules in Equations (8)–(10) for parallel composition are applicable to *labeled transition systems*, it is perfectly feasible to apply them to the composition $P|\beta|Q$ of boolean nets. The only missing elements are transitions labels!

For our investigations, we adopt pairs (P,Q) of nets with identical (n,k) parameters; for the labels, we proceed as follows.

First, we choose the label *alphabet*, which consists of the set $\{1,2\ldots 2nk\}$ of natural numbers (the choice of size $2nk$ is justified below). We overload symbol β to denote both a natural number, with $0 \leq \beta \leq 2nk$, and the set of *synchronization labels* $\{1,2\ldots\beta\}$, so that $P|\{1,2\ldots\beta\}|Q$ is written $P|\beta|Q$. In particular, $\beta = 0$ corresponds to the pure interleaving case $P|||Q$ mentioned in the previous section. As a natural number, β represents the *coupling factor* between P and Q: the larger is β, the more frequent will be the steps in which P and Q must synchronize.

Second, we turn P and Q into *labeled bool nets* by adding two independent functions L_P and L_Q that, respectively, assign a label to each transition $x_P \rightarrow y_P$ and $x_Q \rightarrow y_Q$:

$$L_P, L_Q : \{0,1\}^n \times \{0,1\}^n \rightarrow \{1,2\ldots 2nk\}. \tag{11}$$

Aiming at maximum generality, our labels depend both on the source and on the target state, and are picked at random from set $\{1,2\ldots 2nk\}$.

On this basis, the application of the rules in Equations (8)–(10) to $P|\beta|Q$ becomes possible also when P and Q are labeled bool nets. Note that this can be done regardless of the mode—*sync*, *async* or *hybrid*—in which P and Q are executed.

It is important not to confuse the concept of *sync*/*async* execution mode of P and Q with the (orthogonal) concept of *synchronous*/*asynchronous* transition of $P|\beta|Q$. The execution mode refers to the individual component P or Q, and when we attribute some execution mode to the whole $P|\beta|Q$ we mean that both P and Q operate, *internally*, according to that mode; in principle, we could even imagine composing a P operating in *sync* mode with, e.g., a Q operating in *async* or *hybrid* mode (but in this paper we never do that). On the other hand, a *synchronous* transition of $P|\beta|Q$ is one in which P and Q proceed jointly, each contributing with a local transition performed according to its own mode; furthermore, the two local and simultaneous transitions must have the same label γ, with $\gamma \in \{1,2\ldots\beta\}$. Conversely, an *asynchronous* transition of $P|\beta|Q$ corresponds to a local, γ-labeled transition performed autonomously (and according to its own mode) by only one of the two components, where $\gamma \notin \{1,2\ldots\beta\}$.

4.1. Conditional Dependence in Parallel Composition

We discuss the issue of conditional independence in Section 2, relative to pure bool nets. How does parallel bool net composition $P|\beta|Q$ perform with respect to this property?

The question involves comparing $prob(y_P|x_{PQ}, y_Q)$ with $prob(y_P|x_{PQ})$ where, as before, x and y are states at time t and $t+1$, respectively, and the subscripts identify the relevant system components.

Regardless of the execution mode of the two components, parallel composition *does violate conditional independence*. The reason is that knowing y_Q and finding $y_Q \neq x_Q$ indicates that Q has indeed performed a local transition, whose label, e.g., γ, we can partly or completely deduce from labeling function L_Q, which is known. If $\gamma \notin \{1, 2 \ldots \beta\}$, the system as a whole must have performed an asynchronous (interleaving) transition, in which P must have idled: we immediately deduce $y_P = x_P$. If, conversely, $\gamma \in \{1, 2 \ldots \beta\}$, the system as a whole must have performed a synchronous transition, one in which P has performed a γ-labeled transition jointly with Q: this still tells us something about y_P. In both cases, we acquire more information about y_P than what x_{PQ} alone can give.

In the area of formal methods for Software Engineering, to which Process Algebras belong, it is indeed conditional *dependence* that plays an important role. Consider, for example, the *constraint-oriented specification style* [12,13]. In this style, the parallel composition operator is used as a sort of logical conjunction: system behavior is specified by progressively accumulating constraints (processes) on the ordering of communication events and, possibly, on the exchanged data values. Each constraint reflects a different, partial view on the global system behavior, and all these views should agree on each global transition $x \rightarrow y$. This agreement, governed by the inference rules in Equations (8)–(10), reflects a sort of on-the-fly communication between P and Q, as the global transition occurs. Overall, the effect of those rules is to introduce a mutual dependency among local transitions, which, in terms of conditional mutual information between local state components, means $M(y_P, y_Q | x_{PQ}) \neq 0$.

4.2. Deadlocks

No matter which execution mode is considered, a bool net will always be able to perform transitions from any state. This is not the case for bool net *composition* $P|\beta|Q$, when $\beta > 0$. A *deadlock* occurs at global state $X_{PQ} = X_P.X_Q$, formed by the concatenation of local states X_P and X_Q, when: (i) no a-labeled local transitions are possible from state X_P *or* X_Q, with $a \notin \{1 \ldots \beta\}$ (these would become global, interleaving, asynchronous transitions by the rule in Equation (8) or Equation (9)); and (ii) no pair of local b-labeled transitions is possible from X_P *and* X_Q, with $b \in \{1 \ldots \beta\}$ (yielding global, synchronous transitions by the rule in Equation (10)). In this case, X_{PQ} is a *deadlock state*.

Each *tpm* row should be a probability vector: its total must be 1. However, when X_{PQ} is a deadlock state there is no possible successor Y_{PQ}, and all elements $tpm^m_{P|\beta|Q}(X_{PQ}, *)$ of the corresponding row would be 0s, thus violating the probability vector property. One option sometimes adopted for restoring that property is to set $tpm(X_{PQ}, X_{PQ}) = 1$, forcing the system to permanently remain in that state, and turning a *static* into a *dynamic* deadlock:

static deadlocks —some *tpm* rows, called *null rows*, only have 0s, and are not proper probability vectors;
dynamic deadlocks —all rows are probability vectors, with loop-edges added.

The introduction of dynamic deadlocks preserves the probabilistic nature of the *tpm*, but does not discriminate between actual deadlocks and loop-transitions—those for which the source and target state coincide.

Deadlocks tend to increase as the coupling between the interacting parties becomes stronger:

Proposition 1. *Let P and Q be two labeled bool nets, and $D(P|\beta|Q)$ be the set of deadlock states of system $P|\beta|Q$. Then, $\beta_1 < \beta_2$ implies $D(P|\beta_1|Q) \subseteq D(P|\beta_2|Q)$.*

Proof. We prove by contradiction that if global state x is a deadlock for $P|\beta_1|Q$, it is also a deadlock for $P|\beta_2|Q$. Assume x is not a deadlock for $P|\beta_2|Q$. Then, $P|\beta_2|Q$ can perform (at least) a labeled transition $x \xrightarrow{a} y$.

If $a \in \{1 \ldots \beta_2\}$, the transition is a synchronization between P and Q, supported by the inference rule in Equation (10): then, either $a \in \{1 \ldots \beta_1\}$ or $a \in \{\beta_1 + 1 \ldots \beta_2\}$. In the first case, transition $x \xrightarrow{a} y$ would be feasible also for $P|\beta_1|Q$ (a contradiction); in the second case, the two component transitions $x_P \xrightarrow{a} y_P$ and $x_Q \xrightarrow{a} y_Q$ would enable, by the inference rules in Equations (8) and (9), two global, interleaving transitions of $P|\beta_1|Q$ (a contradiction).

If, on the other hand, $a \notin \{1 \ldots \beta_2\}$, then $x \xrightarrow{a} y$ is an interleaving transition for $P|\beta_2|Q$, which would be a fortiori a feasible interleaving transition for $P|\beta_1|Q$ (a contradiction). $\square$

Furthermore, given composition $P|\beta|Q$, we can establish the following relations among the deadlock sets for the different execution modes.

Proposition 2. *Let P and Q be two labeled bool nets, let $P^{mode}|\beta|Q^{mode}$ be system $P|\beta|Q$ executed in the specified* mode, *and let D be the deadlock set function of Proposition 1. Then, (i) $D(P^{hybrid}|\beta|Q^{hybrid}) \subseteq D(P^{sync}|\beta|Q^{sync})$; and (ii) $D(P^{hybrid}|\beta|Q^{hybrid}) \subseteq D(P^{async}|\beta|Q^{async})$.*

Proof. Part (i). We show by contradiction that, if global state x is a deadlock for $P^{hybrid}|\beta|Q^{hybrid}$, it is also a deadlock for $P^{sync}|\beta|Q^{sync}$. If x were not a deadlock for $P^{sync}|\beta|Q^{sync}$, then this system could escape state x by some transition involving the simultaneous firing of all nodes of N_P *and* N_Q (by the inference rule in Equation (10)), or the firing of all nodes of N_P *or* N_Q (by the inference rule in Equation (8) or Equation (9)). These three firing scenarios are feasible also under the *hybrid* execution mode (see the definitions of the firing group sets FG for the three modes in Section 2), yielding a transition escaping state x also for system $P^{hybrid}|\beta|Q^{hybrid}$—a contradiction.

The proof for Part (ii) is analogous. $\square$

Figure 1 shows the count of deadlock states, out of $2^{5+5} = 1024$ possible states, as a function of the coupling parameter β, for the parallel composition $P^m(5,3)|\beta|Q^m(5,3)$ of two randomly generated, labeled $(5,3)$-bool nets executed in mode m = *sync*, *async* or *hybrid*.

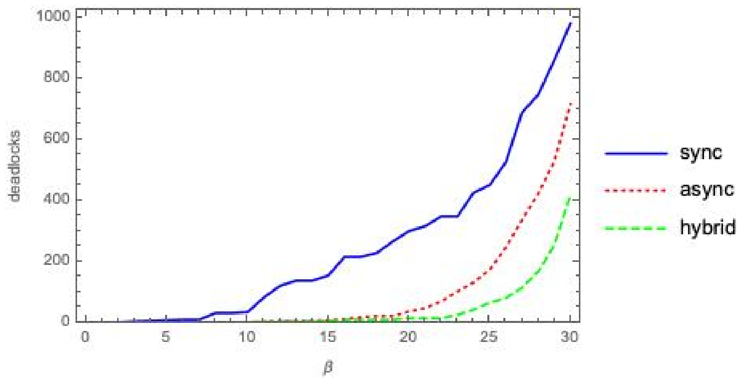

Figure 1. Deadlocks in $P(5,3)|\beta|Q(5,3)$ as a function of coupling factor β, under *sync*, *async* and *hybrid* execution modes.

The plots in Figure 1 provide experimental evidence for Propositions 1 and 2. Indeed, they might also suggest that a deadlock state x for $P^{async}|\beta|Q^{async}$ must also be a deadlock state for $P^{sync}|\beta|Q^{sync}$. However, this is not always the case, as shown by the following simple counterexample.

Assume P and Q are two labeled (n,k)-bool nets with $n = 2$ and $k = 1$. P and Q have identical topology—node 1 reads node 2 and vice versa—and all nodes are associated with the same *bit-flip* bool function. The label set is $\{1,2,3,4\}$, Assume labeling functions L_P and L_Q are defined so that

- $L_P((0,0),(1,1)) = 1, L_P((0,0),(1,0)) = 1, L_P((0,0),(0,1)) = 2$; and
- $L_Q((0,0),(1,1)) = 1, L_Q((0,0),(1,0)) = 3, L_Q((0,0),(0,1)) = 4$.

Then, if we impose maximum synchronization between P and Q, by writing $P|4|Q$, we find that global state $x = (0,0,0,0)$ is a deadlock for $P^{async}|4|Q^{async}$ (since P^{async} can only perform local transitions $(0,0) \xrightarrow{1} (1,0)$ and $(0,0) \xrightarrow{2} (0,1)$ while Q^{async} can only offer $(0,0) \xrightarrow{3} (1,0)$ and $(0,0) \xrightarrow{4} (0,1)$, with no label matching between P and Q) but it is not a deadlock for $P^{sync}|4|Q^{sync}$ (since P^{sync} and Q^{sync} can synchronize by performing local transitions $(0,0) \xrightarrow{1} (1,1)$).

5. Bool Nets as $P{<}\alpha{>}Q$ *sharVar* Compositions

In analogy with the expression $P|\beta|Q$ for the composition of two separate bool nets P and Q by shared transitions (Section 4), we let $P{<}\alpha{>}Q$ denote a *single* bool net whose nodes are partitioned into sets B_P and B_Q, and where there are exactly α *"bridges"*, i.e., directed edges with one endpoint in B_P and the other in B_Q. Bridges allow the two bool net parts—called P and Q—to share and cross-read some of their variables; the other edges are "local" to P or Q. We take α as the *degree of coupling* between P and Q. Furthermore, P^α and Q^α, later equivalently denoted $P*$ and $Q*$, represent the two components *after separation*: a bridge directed from P to Q (or vice versa) turns into a *dangling edge* of Q (or P), with no specified source node (the notation P^α and Q^α is meant to recall the presence of α bridges in the original, uncut bool net; however, it may still happen that one of the components, or both, when $\alpha = 0$, has no dangling edges after separation).

What if we are now given two *independent* bool nets P and Q and we want to *derive* from them some system $P{<}\alpha{>}Q$ with target coupling factor α? This is done by some surgery: we turn α local edges of P and/or Q into bridges between P and Q. The choice of which local edge to turn into a bridge is made at random, and so is the choice of a new source node for it.

Thus, while in building $P|\beta|Q$ the two arguments of the composition are unaffected, except for the addition of the labeling functions L_P and L_Q, for building $P{<}\alpha{>}Q$ we do change the topology of the components, although the node sets B_P and B_Q and the sets of boolean functions F_P and F_Q are preserved. Strictly, ${<}\alpha{>}$ should not be regarded as an algebraic operator, since the operation affects the operands. However, notations $P{<}\alpha{>}Q$ and $P|\beta|Q$ are useful for highlighting the system bipartition and the involved degree of coupling.

Let us now clarify a final, subtle point about execution modes for the two types of cooperation—${<}\alpha{>}$ and $|\beta|$. While expression $P^m|\beta|Q^m$ completely defines the behavior of the system, expression $P^m{<}\alpha{>}Q^m$ *would not*.

In the first case, execution mode m defines the individual behaviors of P^m and Q^m in terms of their possible firing groups, while $|\beta|$ defines the possible transition pairings, i.e., whether or not, given current state X_{PQ}, a firing group of P can fire simultaneously with one of Q, which depends on the involved transition labels. In other words, m defines the firing groups at the local level and $|\beta|$ controls them at the global level, by the mediation of transition labels.

In the second case, the potential firing groups of P^m and Q^m are well defined too, but we have no indication of how they should be combined to yield global transitions: should they act simultaneously or not? The solution is to understand the execution mode as *applied to the net as a whole.* Correspondingly, the correct, unambiguous notation for the *sharVar* cooperation mechanism would be $(P{<}\alpha{>}Q)^m$, although this will often be left implicit.

6. Integrated Information $\bar{\phi}$ for $P{<}\alpha{>}Q$ (*sharVar*)

In very abstract terms, *state-dependent integrated information* $\phi(Y)$, relative to a global system state Y, reduces to the distance or difference d between two Y-dependent probabilistic distributions $x_{coop}(Y)$ and $x_{indep}(Y)$:

$$\phi(Y) = d(x_{coop}(Y), x_{indep}(Y)). \tag{12}$$

Note the slight abuse of notation: $x(Y)$ denotes here, and in similar contexts in the sequel, a distribution x that depends, *as a whole*, on some (state) value Y; $x(X)$ elsewhere is used to select a specific element of distribution x. The meaning should be clear from the context, and is facilitated by our consistent use of symbols x/X and y/Y, for predecessor and successor states. In IIT terminology, $x(Y)$ denotes a *cause repertoire*, as is clear in Equation (15); similarly, $y(X)$ would denote an *effect repertoire*.

Furthermore, $x_{coop}(Y)$, $x_{indep}(Y)$ and consequently $\phi(Y)$ are defined with a system *partition* $\{P, Q\}$ in mind (we restrict to bipartitions) (strictly, ϕ should refer to a specific partition, namely the *Minimum Information Partition* (*MIP*) [1,2], but we apply it to any (bi)partition). Distribution x_{coop}

refers to the system behavior in which parts P and Q *cooperate* according to the relevant interaction mechanism, e.g., *sharVar* or *sharTrans*. With distribution x_{indep} the parts are assumed to operate *independently*. Hence, their difference d is meant to measure the added value provided by cooperation over independent operation.

In IIT 2.0 [1], d is *relative entropy dkl* (Kullback–Leibler divergence):

$$dkl[x_1||x_2] = \sum_{i=1}^{N} x_1(X_i) Log_2 \frac{x_1(X_i)}{x_2(X_i)}, \tag{13}$$

where x_1 and x_2 are two distributions on the same discrete domain $\{X_1 \dots X_N\}$. Note that $dkl[x||x] = 0$: the *dkl* of two equal distributions is null. Note also that, in light of Equation (13), one can express the mutual information in Equation (6) as follows:

$$M(x_1, x_2) = dkl[x_1 x_2 || x_1 \times x_2], \tag{14}$$

where x_1 and x_2 are two random variables with joint distribution denoted $x_1 x_2$ and respective marginal distributions x_1 and x_2 (we hope symbol overloading is no too confusing here!). Symbol "$\times$" in Equation (14) denotes distribution product, which is defined in the main text.

Consider an (n,k)-bool net $(P{<}\alpha{>}Q)^m$ executed in mode m (*sync*, *async* or *hybrid*), denoted PQ for short. The behavior of the net is fully defined by the transition probability matrix tpm^m_{PQ}. It is easy to see that, regardless of the mode m, tpm^m_{PQ} cannot have *null-rows* (all 0s), corresponding to deadlocks. Note, however, that one may find *null-columns* with the *sync* and *async* modes, corresponding to "Garden of Eden" states Y_{PQ} that have no predecessor state X_{PQ}. This does not happen with the *hybrid* mode, since the firing groups for this mode include the *empty firing group* (no node fires), which creates a loop-edge: any state has itself as a predecessor.

For the subsequent definitions of integrated information for PQ, we also need tpm^m_{P*} and tpm^m_{Q*}: these are the tpms that characterize the *independent* behaviors, under mode m, of $P*$ and $Q*$, i.e., the two components P and Q after separation, when the data flowing across the α bridges from one to the other are lost due to the cut, and replaced by *white noise*, i.e., uniformly distributed bit tuples.

We are finally ready to actualize the abstract definition of Equation (12) into the concrete definition given in [1]. The state-dependent integrated information $\phi^m_{P<\alpha>Q}(Y_{PQ})$ for global state Y_{PQ} of (n,k)-bool net $PQ = P{<}\alpha{>}Q$ executed in mode m is:

$$\phi^m_{P<\alpha>Q}(Y_{PQ}) = dkl[pre^m_{P<\alpha>Q}(Y_{PQ}) || pre^m_{P*}(Y_P) \times pre^m_{Q*}(Y_Q)] \tag{15}$$

where:

- $pre^m_{P<\alpha>Q}(Y_{PQ})$ is the distribution of the *predecessors* of state Y_{PQ}, obtained by normalizing $tpm^m_{P<\alpha>Q}(*, Y_{PQ})$—the Y_{PQ}-indexed column of $tpm^m_{P<\alpha>Q}$;
- Y_P and Y_Q are the P and Q components of state Y_{PQ}: $Y_{PQ} = Y_P.Y_Q$ (concatenation);
- $pre^m_{P*}(Y_P)$ and $pre^m_{Q*}(Y_Q)$ are the distributions of the predecessors of, respectively, Y_P and Y_Q, obtained as done for $pre^m_{P<\alpha>Q}(Y_{PQ})$ but using, respectively, tpm^m_{P*} and tpm^m_{Q*}; and
- "$\times$" is distribution multiplication: if d_1 and d_2 are probability distributions defined, respectively, over $\{0,1\}^{n1}$ and $\{0,1\}^{n2}$—the sets of tuples of lengths $n1$ and $n2$—and $d = d_1 \times d_2$ is the distribution product, then, for the generic $(n1+n2)$-bit tuple $X^{n1+n2} = X^{n1}.X^{n2}$, we have $d(X^{n1+n2}) = d(X^{n1}) * d(X^{n2})$.

(The interested reader can find in [14] a freely downloadable demonstration tool illustrating state-dependent ϕ for bool nets executed in the standard, *sync* mode, for generic partitions.)

The averaged form $\bar{\phi}^{m}_{dkl}(P{<}\alpha{>}Q)$ of integrated information (subscript "*dkl*" is convenient in light of subsequent developments) is defined as a weighted sum over all states Y_{PQ} of the state dependent $\phi^{m}_{P<\alpha>Q}(Y_{PQ})$s—a weighted sum that we conveniently express as a dot product ("."):

$$\bar{\phi}^{m}_{dkl}(P{<}\alpha{>}Q) = post^{m}_{P<\alpha>Q}(\bar{x}_{PQ}).Table[\phi^{m}_{P<\alpha>Q}(Y_{PQ})|(Y_{PQ})_{10} = 0,1\ldots 2^{n}-1] \tag{16}$$

where:

- $\bar{x}_{PQ}$ denotes the uniform distribution of PQ states (n-tuples of bits);
- $post^{m}_{P<\alpha>Q}(\bar{x}_{PQ})$, expressing the weights of the sum, is the distribution of the *successors* of *state distribution* $\bar{x}_{PQ}$. Note that we conceive functions *pred* and *post* to be applicable both to a specific state (some bit tuple X or Y) and to a distribution of such states, e.g., to $\bar{x}_{PQ}$. No ambiguity arises, since we always use lowercase to denote random variables or their distributions (x and y), and uppercase to denote specific state values (X and Y). (Using a distribution as argument of *pred* or *post* is preferred, since a specific state, e.g., state $\{0,0,1\}$ of a three-bit bool net, can be represented as distribution $\{0,1,0,0,0,0,0,0\}$ assigning probability 1 to the second triple of bits, when these are presented in lexicographic order, and probability 0 to all other triples. In particular, $post^{m}_{PQ}(\bar{x}_{PQ}) = \bar{x}_{PQ}.tpm_{PQ}$.)
- $(Y_P)_{10}$ is the decimal representation of bit tuple Y_P.
- $Table[\ldots]$ is the list of $\phi^{m}_{P<\alpha>Q}(Y_{PQ})$ values for all bit n-tuples Y_{PQ}, listed in lexicographic order.

In [15], it is shown that $\bar{\phi}_{dkl}(P{<}\alpha{>}Q)$ can also be computed as $M(\bar{x}_{PQ}, y_{PQ}) - [M(\bar{x}_{P*}, y_{P*}) + M(\bar{x}_{Q*}, y_{Q*})]$, where the barred symbols denote uniform distributions (maximum entropy), and M is mutual information between current and next state, both referred to the global system PQ and to the two noised components $P*$ and $Q*$. In our experiments, we took advantage of this alternative definition, which is computationally more efficient.

6.1. The dkl*-Mismatch Problem for $P{<}\alpha{>}Q$*

In light of its definition in Equation (13), $dkl[x_1||x_2]$ is *undefined* when $x_1(X_i) > 0$ and $x_2(X_i) = 0$ for some state X_i: this is what we call the "*dkl-mismatch problem*".

Proposition 3 establishes that this problem does not arise with the $\bar{\phi}^{m}_{dkl}(P{<}\alpha{>}Q)$ we are considering, at least relative to two execution modes.

Proposition 3. *Given a partitioned bool net $P{<}\alpha{>}Q$, the dkl-mismatch problem does not arise for $\bar{\phi}^{m}_{dkl}(P{<}\alpha{>}Q)$, when mode m is* sync *or* hybrid.

Proof. In light of the definition in Equation (15), we must prove that, when an element of distribution $pre^{m}_{P<\alpha>Q}(Y_{PQ})$ is different from zero, so is the corresponding element of distribution $pre^{m}_{P*}(Y_P) \times pre^{m}_{Q*}(Y_Q)$. For notational convenience, let these two distributions be called, respectively, x_1 and x_2, as in Equation (13). Then, we must prove that, for any state X_{PQ}: $x_1(X_{PQ}) > 0$ implies $x_2(X_{PQ}) > 0$, in *sync* and *hybrid* mode. Now, $x_1(X_{PQ}) > 0$ means that there exists (at least) one transition

$$X_{PQ} \xrightarrow{fg^{m}_{PQ}} Y_{PQ}$$

triggered by some firing group fg^{m}_{PQ}. Correspondingly, $tpm^{m}_{P<\alpha>Q}(X_{PQ}, Y_{PQ}) > 0$.

Representing firing groups as node sets, and observing that the firing groups of the whole system $P{<}\alpha{>}Q$ and of its parts are independent from α, we can take advantage of Equations (1)–(3), which refer to $P \cup Q \equiv P{<}0{>}Q$, finding that a global firing group can be decomposed into two local firing groups, under the same mode—$fg^{m}_{PQ} = fg^{m}_{P} \cup fg^{m}_{Q}$—only for $m = sync$ and $m = hybrid$ (for $m = async$, fg^{async}_{PQ} must include exactly *one* node, while any fg^{async}_{P} and any fg^{async}_{P} must include one node each, so that $fg^{async}_{P} \cup fg^{async}_{P}$ includes *two*).

As a consequence, using a functional notation for transitions, for $m = sync$ and $m = hybrid$ we can write:

$$Y_{PQ} = Y_P.Y_Q = fg^m_{PQ}(X_{PQ}) = fg^m_P(X_{PQ}).fg^m_Q(X_{PQ}) = fg^m_P(X_P.X_Q).fg^m_Q(X_P.X_Q).$$

The fact that $Y_P = fg^m_P(X_P.X_Q)$ guarantees that $tpm^m_{P*}(X_P, Y_P) > 0$: by definition, element $tpm^m_{P*}(X_P, Y_P)$ of "noised" matrix tpm^m_{P*} is obtained by the cumulative contribution of all values $tpm^m_{PQ}(X_P.*, Y_P.*)$, and we have assumed above that at least one of them, namely $tpm^m_{P<\alpha>Q}(X_P.X_Q, Y_P.Y_Q)$, gives a non-null contribution. Similarly, we find that $tpm^m_{Q*}(X_Q, Y_Q) > 0$. We conclude that the cut sub-systems $P*$ and $Q*$ separately support transitions $X_P \to Y_P$ and $X_Q \to Y_Q$, meaning that distribution $pre^m_{P*}(Y_P)$ (respectively, $pre^m_{Q*}(Y_Q)$) assigns a non-null probability to its X_P-indexed (respectively, X_Q-indexed) element. Since x_2 was defined as $pre^m_{P*}(Y_P) \times pre^m_{Q*}(Y_Q)$, we conclude that $x_2(X_P.X_Q) > 0$. $\square$

The "anomaly" of the *async* mode already observed in Equations (1)–(3), and in Section 2, is further highlighted in Proposition 3. For these reasons, we drop this execution mode, and in the sequel m is only *sync* or *hybrid*.

The computation of $\bar{\phi}^m_{dkl}(P{<}\alpha{>}Q)$ is not even affected by the presence of "Garden of Eden" states. If Y'_{PQ} is such a state, we might perhaps represent the predecessor distribution $pre^m_{P<\alpha>Q}(Y'_{PQ})$ as the null "probability vector"; then, regardless of the second argument $pre^m_{P*}(Y'_P) \times pre^m_{Q*}(Y'_Q)$ of dkl, we would obtain $\phi^m_{P<\alpha>Q}(Y'_{PQ}) = 0$. However, in any case, these null values are selected away in the weighted sum of the definition in Equation (16), since state distribution $post^m_{P<\alpha>Q}(\bar{x}_{PQ})$—providing the weights—assigns probability 0 to Y'_{PQ} since, by the definition of Garden of Eden state, none of the X_{PQ}s can transition to the latter.

6.2. Statistical Results for $\bar{\phi}^m_{dkl}(P{<}\alpha{>}Q)$

Having defined $\bar{\phi}^m_{dkl}(P{<}\alpha{>}Q)$, we wish to investigate the dependence of this measure on α, the degree of coupling between P and Q when they cooperate by shared variables.

Letting $n = 5$ and $k = 3$, we have built ten pairs $(P_i(n,k), Q_i(n,k))$, $i = 1 \ldots 10$, of randomly generated (n,k)-bool nets and have derived, for each pair, the sequence of $P_i{<}\alpha{>}Q_i$ systems for values $\alpha = 0, 1 \ldots 2nk$ (with $2nk = 30$), as described at the beginning of Section 5. Then, for each α, we have computed $Mean^{10}_{i=1}\{\bar{\phi}^m_{dkl}(P_i{<}\alpha{>}Q_i)\}$ and associated standard deviation, for $m = sync$ and $hybrid$. The results of the simulation are shown in the plots of Figure 2.

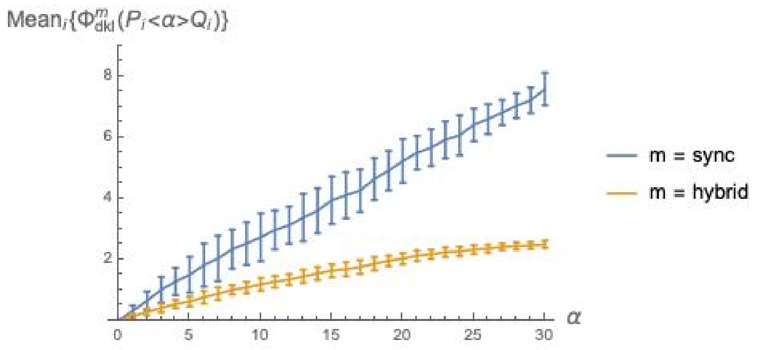

Figure 2. Mean values of $\bar{\phi}^m_{dkl}(P_i{<}\alpha{>}Q_i)$, $i = 1 \ldots 10$, as a function of the coupling factor α, for execution modes *sync* and *hybrid*.

The plots in Figure 2 confirm the intuitive expectation that integrated information grows with the coupling factor α between P and Q. Recall that $\bar{\phi}$ is a weighted sum of $\phi(Y_{PQ})s$ (Equation (16)), and that $\phi(Y_{PQ})$ is a dkl "distance" between an x_{coop} and an x_{indep} distribution (Equation (15)). As α grows, it is to be expected that x_{coop} and x_{indep} drift apart, since a larger α means a stronger mutual influence between the behaviors of P and Q, thus a more marked departure from the behaviors they exhibit when acting independently.

The fact that $\bar{\phi}^{sync}_{dkl} > \bar{\phi}^{hybrid}_{dkl}$ can be intuitively explained as follows. Due to the different sizes of the involved firing group sets—$|FG^{sync}_{PQ}| = 1$ and $|FG^{hybrid}_{PQ}| = 2^{2n}$—in *sync* mode the *successor* distribution $post^{sync}_{PQ}(X_{PQ})$ is "punctual" (one element has probability 1, the others have probability 0), while $post^{hybrid}_{PQ}(X_{PQ})$ is much more spread over the states. An analogous difference in spread can be observed also when looking backward, with *predecessor* distributions $pre^{sync}_{PQ}(Y_{PQ})$ and $pre^{hybrid}_{PQ}(Y_{PQ})$. The local *post* and *pre* distributions for P and Q follow a similar pattern with respect to *sync* vs. *hybrid*. Then, in

$$dkl[pre^{m}_{P<\alpha>Q}(Y_{PQ})||pre^{m}_{P*}(Y_P) \times pre^{m}_{Q*}(Y_Q)],$$

the two argument distributions are closer to each other for $m = hybrid$ than for $m = sync$, due to the higher spread of the distributions for the *hybrid* mode. Note that the local distributions are also affected by noise injection, which cannot but amplify their spread, pushing them closer to the global distribution with higher spread, namely the distribution for the *hybrid* mode, and smaller distance means smaller $\bar{\phi}$.

6.3. Statistical Results for $\bar{\phi}^{m}_{Manh}(P<\alpha>Q)$ Using Manhattan Distance

We show that, for the *sharTrans* cooperation mechanism $|\beta|$, the *dkl*-mismatch problem becomes pervasive. Thus, for enabling comparisons between *sharVar* and *sharTrans* cooperation in terms of integrated information, we consider also a state-dependent version $\hat{\phi}$ of this measure in which the *dkl* of Equation (15) is replaced by Manhattan distance (*Manh*):

$$\hat{\phi}^{m}_{P<\alpha>Q}(Y_{PQ}) = Manh(pre^{m}_{P<\alpha>Q}(Y_{PQ}), pre^{m}_{P*}(Y_P) \times pre^{m}_{Q*}(Y_Q)). \tag{17}$$

$\hat{\phi}^{m}_{P<\alpha>Q}(Y_{PQ})$ is then used, as in Equation (16), for obtaining the corresponding state-independent $\bar{\phi}^{m}_{Manh}(P<\alpha>Q)$:

$$\bar{\phi}^{m}_{Manh}(P<\alpha>Q) = post^{m}_{P<\alpha>Q}(\bar{x}_{P<\alpha>Q}).Table[\hat{\phi}^{m}_{P<\alpha>Q}(Y_{PQ})|(Y_{PQ})_{10} = 0, 1 \ldots 2^n - 1]. \tag{18}$$

We have conducted a statistical analysis for $\bar{\phi}^{m}_{Manh}(P<\alpha>Q)$ analogous to that presented in Section 6.2. The results are illustrated in Figure 3. This figure indicates that Manhattan distance broadly agrees with *dkl* (while not suffering from the mismatch problem), and confirms the two general facts already established with Figure 2: integrated information grows with the coupling factor α, and is higher for the *sync* than for the *hybrid* execution mode.

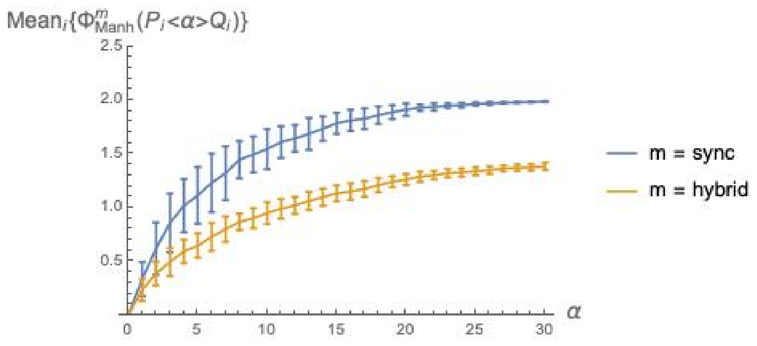

Figure 3. Mean values of $\bar{\phi}^{m}_{Manh}(P_i<\alpha>Q_i)$, $i = 1 \ldots 10$, as a function of the coupling factor α, for execution modes *sync* and *hybrid*, using *Manhattan distance* in place of *dkl*.

7. Integrated Information $\bar{\phi}$ for $P|\beta|Q$ (LOTOS *sharTrans*)

How can one define integrated information $\bar{\phi}$ for *sharTrans* composition $P|\beta|Q$? The problem reduces to one of adapting to the new context the state-dependent measure $\phi(Y)$ of Equation (12), which is given a concrete form in Equation (15).

In fact, in $dkl(x_{coop}(Y), x_{indep}(Y))$, the first argument is readily defined even in the new setting, since it refers to the "cooperative" behavior of $P|\beta|Q$ in which P and Q interact as specified by operator $|\beta|$—a behavior that is fully defined by $tpm^m_{P|\beta|Q}$—which is, in turn, fully defined by the inference rules in Equations (8)–(10). The difficulty arises with the definition of x_{indep}: what does it mean, in the *sharTrans* context, for P and Q to *operate independently*?

Our proposed solution to this problem stems from the observation that, while under *sharVar* the cooperating parts share knowledge about each other's variables, under *sharTrans* they share *knowledge about the order of transitions in time*, since each part must follow the ordering of transitions of the other, at least limited to the transitions whose labels are in the synchronization set β. We must then conclude that the *absence* of cooperation occurs when there is no shared knowledge about local transition ordering, and no concern to agree on it. This immediately suggests to identify independent behavior—and x_{indep} in expression $dkl(x_{coop}, x_{indep})$—with the pure interleaving composition $P|||Q$ (see Section 3). The resulting definition of state-dependent integrated information for the $|\beta|$ mechanism is then:

$$\phi^m_{P|\beta|Q}(Y_{PQ}) = dkl[pre^m_{P|\beta|Q}(Y_{PQ})||pre^m_{P|||Q}(Y_{PQ})]. \tag{19}$$

Note that the above definition is conceptually (and computationally) simpler than the corresponding definition for $<\alpha>$ (Equation (15)): no input noise for the cut components ($P*$ and $Q*$) is involved, and the second argument of dkl is not a distribution product but simply the first element of the sequence $P|\beta|Q$ for $\beta = 0, 1 \ldots$.

Then, the definitions of *state-independent* integrated information $\bar{\phi}$ for $P|\beta|Q$ and for $P{<}\alpha{>}Q$ are essentially the same (compare with Equation (16)):

$$\bar{\phi}^m_{dkl}(P|\beta|Q) = post^m_{P|\beta|Q}(\bar{x}_{PQ}).Table[\phi^m_{P|\beta|Q}(Y_{PQ})|(Y_{PQ})_{10} = 0, 1 \ldots 2^{2n} - 1]. \tag{20}$$

Note that n is now the number of nodes in P, which has the same size as Q, yielding a total of $2n$ nodes.

7.1. The dkl-*Mismatch Problem for* $P|\beta|Q$

As anticipated, the *dkl*-mismatch problem becomes pervasive with systems of type $P|\beta|Q$. Let state Y_{PQ} be fixed and consider distributions $x_1 = pre^m_{P|\beta|Q}(Y_{PQ})$ and $x_2 = pre^m_{P|||Q}(Y_{PQ})$ in Equation (19). The mismatch occurs when $x_1(X'_{PQ}) > 0$ and $x_2(X'_{PQ}) = 0$ for some X'_{PQ}. In *sync* mode, it is easy to imagine a transition $X'_{PQ} \xrightarrow{P|\beta|Q} Y_{PQ}$ of system $P|\beta|Q$ such that the two bit tuples X'_{PQ} and Y_{PQ} differ both in their P and in their Q component: this may happen when the transition is a synchronization. Since X'_{PQ} is a predecessor state of Y_{PQ}, we have $x_1(X'_{PQ}) > 0$. On the contrary, no predecessor of Y_{PQ} under system $P|||Q$ can differ from Y_{PQ} in both state components, since system $P|||Q$ must fire one component at a time, as explained at the end of Section 3. Thus, necessarily, $x_2(X'_{PQ}) = 0$: this yields the mismatch. The argument for the *hybrid* mode is analogous, the key point being that a firing group of $P|\beta|Q$ may involve nodes from both P and Q, while a firing group of $P|||Q$ involves nodes exclusively from one component, by the definition of "$|||$".

To give an idea of how severe the *dkl*-mismatch problem is for $P|\beta|Q$, we have counted the number of states Y_{PQ} yielding a *dkl*-mismatch for each of the 310 systems $(P_i{<}\beta{>}Q_i)$, for $i = 1 \ldots 10$ and $\beta = 0 \ldots 30$, where the (P_i, Q_i) pairs are those already used in Section 6 (Figure 2). These numbers are collected in the 10×31 grey-level matrix of Figure 4.

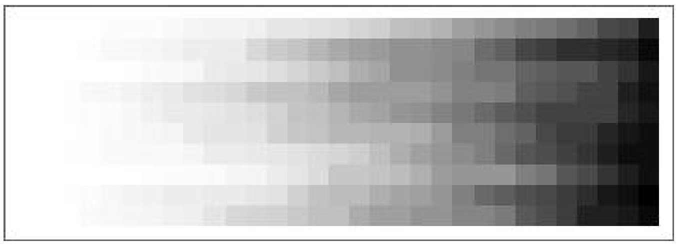

Figure 4. Number of states, represented as grey levels, that give rise to a dkl-mismatch for systems $P_i^{sync}{<}\beta{>}Q_i^{sync}$, $i = 1\dots10$ (rows), $\beta = 0\dots30$ (columns). Systems have up to 993 states, out of 1024, that yield mismatch (the darkest cells), and only 16 systems, out of 310, have none.

7.2. Statistical Results for $\bar{\phi}^m_{Manh}(P|\beta|Q)$ Using Manhattan Distance

In light of the impossibility to use the dkl-based definition of ϕ (Equation (19)) for $|\beta|$ cooperation, we switch, again, to a "hat" version $\hat{\phi}$ in which dkl is replaced by *Manhattan distance*:

$$\hat{\phi}^m_{P|\beta|Q}(Y_{PQ}) = Manh(pre^m_{P|\beta|Q}(Y_{PQ}), pre^m_{P|||Q}(Y_{PQ})), \tag{21}$$

and use it, in turn, for defining the state-independent $\bar{\phi}^m_{Manh}(P|\beta|Q)$, as in Equation (20):

$$\bar{\phi}^m_{Manh}(P|\beta|Q) = post^m_{P|\beta|Q}(\bar{x}_{PQ}).Table[\hat{\phi}^m_{P|\beta|Q}(Y_{PQ})|(Y_{PQ})_{10} = 0, 1\dots2^{2n} - 1]. \tag{22}$$

Analogous to Figure 3, in Figure 5, we plot the values of $\bar{\phi}^m_{Manh}(P|\beta|Q)$ as a function of the coupling factor β, each point obtained by averaging over 10 (P, Q) pairs, both using static deadlocks (Figure 5, left) and dynamic deadlocks (Figure 5, right).

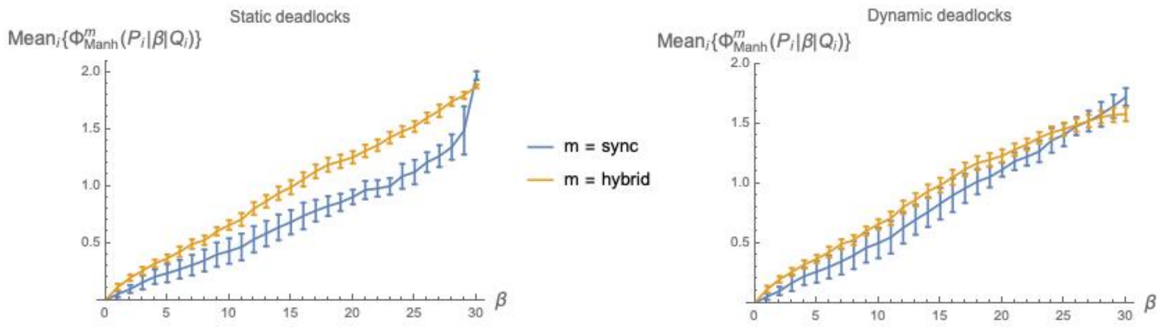

Figure 5. Mean values of $\bar{\phi}^m_{Manh}(P_i|\beta|Q_i)$, $i = 1\dots10$, as a function of the coupling factor β, for execution modes *sync* and *hybrid*, using *Manhattan distance* in place of dkl, using: static deadlocks (**left**); and dynamic deadlocks (**right**).

The distinction between static and dynamic deadlocks was introduced in Section 4.2. Note that when using static deadlocks—no 1s added on the diagonal of $tpm^m_{P|\beta|Q}$—the weights $post^m_{P|\beta|Q}(\bar{x}_{PQ})$ in Equation (22) will in general not total 1, *and must be re-normalized*.

8. Integrated Information $\bar{\phi}$ for $P[\gamma]Q$ (CCS *sharTrans*)

Kullback–Leibler divergence dkl is a central element of Integrated Information Theory 2.0 [1], thua it is indeed desirable to apply it in the new *sharTrans* context without incurring the dkl-mismatch problem. In this section, we propose a slightly different version of *sharTrans* cooperation, directly inspired to Robin Milner's seminal process algebra *CCS* (Calculus of Communicating Systems) [5], that precisely avoids that problem.

Consider the abstract expression:

$$dkl[pre^m_{coop}(Y_{PQ})||pre^m_{indep}(Y_{PQ})].$$

How can we conceive the cooperative $PcoopQ$ and independent $PindepQ$ behaviors of bipartite system PQ (as defined by matrices tpm^m_{coop} and tpm^m_{indep}, from which $pre^m_{coop}(Y_{PQ})$ and $pre^m_{indep}(Y_{PQ})$ are derived) so that the dkl-mismatch problem is ruled out?

Clearly, a *sufficient condition* for avoiding dkl-mismatches is the following:

The existence of a transition $X \xrightarrow{PcoopQ} Y$ *implies* the existence of transition $X \xrightarrow{PindepQ} Y$ between the same states.

The two CCS behavioral operators of (non-parametric) *parallel composition* ("$P|Q$") and (parametric) *restriction* ("$\backslash\gamma$"), where γ is a label set $\{1, 2 \dots \gamma\}$ (using the same convention as for β), offer us a way to define cooperation and independence so that they satisfy the above condition.

In CCS, symbol τ denotes a special *internal, not observable* transition label: no synchronization is possible with a process that performs a τ-labeled transition. Let A be the set of observable labels and define $A^+ = A \cup \{\tau\}$. Symbol a ranges in A and symbol x ranges in A^+. We provide below the four inference rules of the Structural Operational Semantics of CCS for the two mentioned operators (we depart from the standard definition of Milner [5] only in one aspect: we drop the idea of a synchronization based on the matching between a label a and its corresponding "co-label" $\bar{a}$, and revert to the LOTOS requirement that the two labels be simply equal).

$$\frac{P \xrightarrow{x} P' \wedge x \in A^+}{P|Q \xrightarrow{x} P'|Q} \quad (\textit{CCS left interleaving}) \tag{23}$$

$$\frac{Q \xrightarrow{x} Q' \wedge x \in A^+}{P|Q \xrightarrow{x} P|Q'} \quad (\textit{CCS right interleaving}) \tag{24}$$

$$\frac{P \xrightarrow{a} P' \wedge Q \xrightarrow{a} Q' \wedge a \in A}{P|Q \xrightarrow{\tau} P'|Q'} \quad (\textit{CCS synchronization}) \tag{25}$$

$$\frac{P \xrightarrow{a} P' \wedge a \notin \gamma}{P\backslash\gamma \xrightarrow{a} P'} \quad (\textit{CCS restriction}) \tag{26}$$

The "interleaving" rules in Equations (23) and (24) establish that any transition that P or Q could perform *locally*—in itself—can be performed *globally* by composite system $P|Q$. Additionally, the rule in Equation (25) establishes that parallel composition $P|Q$ can also perform synchronization transitions whenever two equally labeled observable transitions are available at the two sides. The rule in Equation (26) defines the restriction $P\backslash\gamma$ as a filter that enables P (which can be itself a two-process composition) to perform a transition only if its label is not in the specified set γ of *forbidden* labels (thus, τ is always admitted), pruning away all other transitions.

We now combine CCS *parallel composition* and *restriction* into the convenient syntactic form

$$P[\gamma]Q \equiv (P|Q)\backslash\gamma, \tag{27}$$

where parameter γ is enclosed in square brackets to distinguish it from the LOTOS form "$|\beta|$", and use it for actualizing the cooperation and independence relations between P and Q:

$$PindepQ = P|Q \equiv P[0]Q \tag{28}$$
$$PcoopQ = P[\gamma]Q. \tag{29}$$

Note that no deadlock can occur in $P|Q$. With the LOTOS-based *sharTrans* composition, we had assumed $PindepQ = P|||Q$, a form of independence by which P and Q will never deadlock *and* will *never* synchronize: their respective transitions can only interleave. This is not the case for the CCS-based approach, where $PindepQ = P[0]Q = P|Q$: the two independent systems, by the rule in Equation (25), can indeed synchronize any pair of transitions $P \xrightarrow{a} P'$ and $Q \xrightarrow{a} Q'$ with the same observable label.

However, by the rules in Equations (23) and (24), these same transitions can be executed separately, in interleaving—in *independence*. Cooperation $P[\gamma]Q \equiv (P|Q)\backslash\gamma$, then, consists in ruling out these independent transitions—at least those specified in set γ—while preserving their synchronizations.

One could argue that $P|Q$ already entails a sort of cooperation, via all the synchronization transitions it supports, and that pure LOTOS interleaving $P|||Q$ is a more appropriate form of independence. This is true only in part. The "cooperation" that takes place in $P|Q$ is *not private*: the transition $P \xrightarrow{a} P'$ that P shares with Q, forming a τ-labeled synchronization, is also "offered" separately by P (and by Q too) for further two-way synchronizations with other potential partners. When we apply restriction to the composition—$(P|Q)\backslash\gamma$—we rule out this possibility, and cooperation via joint transitions with labels in γ becomes exclusive of the (P, Q) pair, occurring via a global, τ-labeled transition. In other words, in $P|Q$, the parties are not *forced* to wait for each other at specific transitions, as in $P|\beta|Q$, while this effect of mutual influence on transition ordering is enforced in $P[\gamma]Q$, when restriction is in action.

It is clear that the above sufficient condition for ruling out *dkl*-mismatches is satisfied by our newly adopted CCS-based definitions:

$$\phi^m_{P[\gamma]Q} \quad = \quad dkl[pre^m_{P[\gamma]Q}(Y_{PQ})||pre^m_{P[0]Q}(Y_{PQ})] \tag{30}$$

$$\bar{\Phi}^m_{dkl}(P[\gamma]Q) \quad = \quad post^m_{P[\gamma]Q}(\bar{x}_{PQ}).Table[\phi^m_{P[\gamma]Q}(Y_{PQ})|(Y_{PQ})_{10} = 0, 1 \ldots 2^{2n} - 1] \tag{31}$$

since, by the definition of the restriction operator, the transitions of $P[\gamma]Q$ are a subset of those of $P[0]Q$.

8.1. Deadlocks in $P[\gamma]Q$

While deadlocks can never occur in $P|Q$, they may occur in $P[\gamma]Q \equiv (P|Q)\backslash\gamma$: this happens when P and Q offer disjoint sets of labels, thus preventing any synchronization between them, and when all these labels are members of γ, the set of forbidden labels.

In Figure 6, we show the count of deadlock states, out of 1024 possible states, as a function of the coupling parameter γ, for the composition $P^m(5,3)[\gamma]Q^m(5,3)$ of two randomly generated labeled (5,3)-bool nets executed in modes *sync* and *hybrid*.

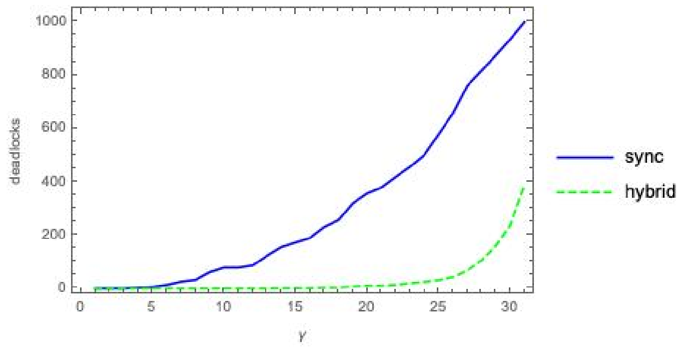

Figure 6. Deadlocks in $P(5,3)|\gamma|Q(5,3)$ as a function of coupling factor γ, under *sync* and *hybrid* execution modes.

Recall that we have dropped the *async* mode for its various anomalies. For the remaining two modes, deadlocks under the $[\gamma]$ and $|\beta|$ interaction mechanisms seem to behave quite similarly (compare with Figure 1).

8.2. Statistical Results for $\bar{\Phi}^m_{dkl}(P[\gamma]Q)$

Before presenting the plots, we need to deal again with static vs. dynamic deadlocks (Section 4.2). Referring to the *sync* mode, *tpms* with dynamic deadlocks turn out to be inappropriate for computing $\bar{\Phi}^{sync}_{dkl}(P[\gamma]Q)$, since they would re-introduce the *dkl*-mismatch problem that we have managed to rule out by switching to the $[\gamma]$ cooperation mechanism! The reason is that a static deadlock is made dynamic by adding a "1" on the diagonal of $tpm^{sync}_{P[\gamma]Q}$, at an otherwise null row. This entry is unlikely to find a non-zero counterpart in $tpm^{sync}_{P[0]Q}$, since $P[0]Q$ (i.e., $P|Q$) has no deadlocks—no 1s added on the diagonal—and the only possibility to have a non-zero entry on the diagonal is that an actual loop-transition $X_{PQ} \to X_{PQ}$ be possible for that system. However, when P and Q are executed in *sync* mode, this is unlikely, both when they operate in interleaving (i.e., when only one of them updates all its nodes) and, even worse, when they synchronize (i.e., when all nodes of P and Q are updated). Thus, for the *sync* mode, the option is to use static deadlocks—no 1s added on the diagonal of $tpm^{m}_{P[\gamma]Q}$. As observed above for the $|\beta|$ composition, the weights $post^{m}_{P[\gamma]Q}(\bar{x}_{PQ})$ in Equation (31), will in general not total 1, and must be re-normalized.

The dkl-mismatch problem does not arise with the *hybrid* mode since, using the empty firing group, a loop edge $X_{PQ} \to X_{PQ}$ is always possible for system $P|Q$, for any state X_{PQ}, so the elements on the diagonal of $tpm^{hybrid}_{P[0]Q}$ are all different from 0. In this case, we can then safely use both dynamic deadlocks and static deadlocks with re-normalization.

In Figure 7, we plot the values of $\bar{\phi}^{m}_{dkl}(P[\gamma]Q)$ as a function of the coupling factor γ, each point obtained by averaging over 10 (P,Q) pairs, using static deadlocks (for the *sync* and *hybrid* modes) and dynamic deadlocks (only for the *hybrid* mode).

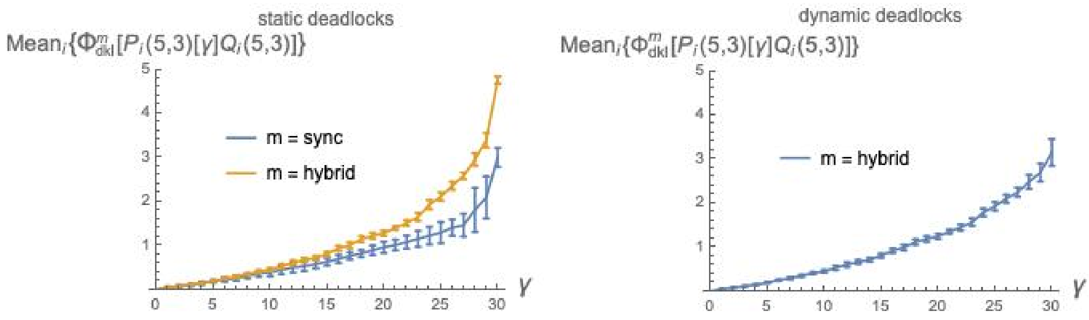

Figure 7. Mean values of $\bar{\phi}^{m}_{dkl}(P_i[\gamma]Q_i)$, $i = 1\ldots10$, as a function of the coupling factor γ, for execution modes *sync* and *hybrid*, using the definitions in Equations (30) and (31). The used *tpm*s implement: static deadlocks (**left**); and dynamic deadlocks (**right**).

8.3. Statistical Results for $\bar{\phi}^{m}_{Manh}(P[\gamma]Q)$ Using Manhattan Distance

The potential mismatch between distributions d_1 and d_2 in $dkl[d_1||d_2]$ is not a concern when using Manhattan distance $Manh(d_1,d_2)$ for defining $\bar{\phi}^{m}_{Manh}(P[\gamma]Q)$ (by equations analogous to Equations (30) and (31)). Thus, we can handle both static deadlocks, with renormalization of the weights, and dynamic deadlocks.

Figure 8 is analogous to Figure 7, except that Manhattan distance is used in place of dkl.

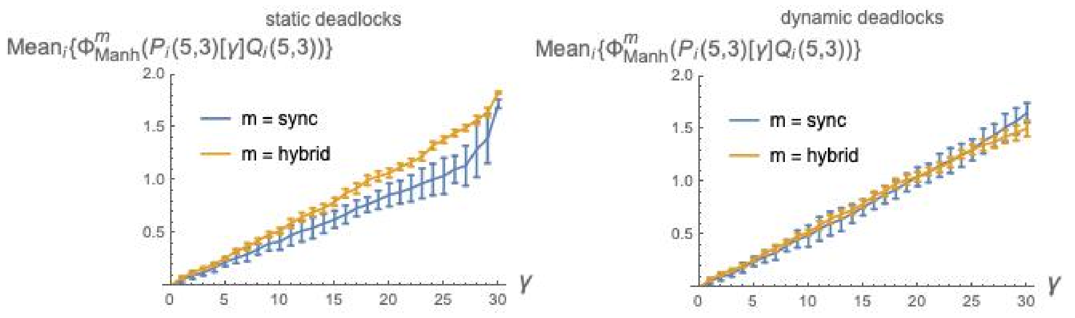

Figure 8. Mean values of $\bar{\phi}^{m}_{Manh}(P_i[\gamma]Q_i)$, $i = 1\ldots10$, as a function of the coupling factor γ, for execution modes *sync* and *hybrid*, using: static deadlocks (**left**); and dynamic deadlocks (**right**).

9. Comparisons

Our experimental analysis has involved three dimensions, or degrees of freedom:

- (i) bool net cooperation mechanism ($\alpha/\beta/\gamma$);
- (ii) bool net execution mode (*sync*/*hybrid*); and
- (iii) "distance" function for probabilistic distributions (*dkl*/*Manhattan distance*).

As stated initially, our interest is primarily in the comparison of cooperation mechanisms (i). It is then useful to aggregate the statistical data collected in the previous sections so that the plots for α, β and γ appear in the same diagram. This is done in Figure 9 which shows, for each fixed choice of execution mode (the columns) and distance function (the rows), the "performance" of the applicable mechanisms in terms of integrated information, as a function of the coupling factor ("coup"). Note that Manhattan distance is represented in Rows 2 *and* 3, corresponding to systems implementing, respectively, static and dynamic deadlocks: this distinction only affects the β and γ plots, since the α mechanism is immune to deadlocks. For convenience, the α plots in Row 2 are replicated in Row 3. (Recall also that the *dkl*-mismatch problem prevented us from applying distance *dkl* to the β mechanism.)

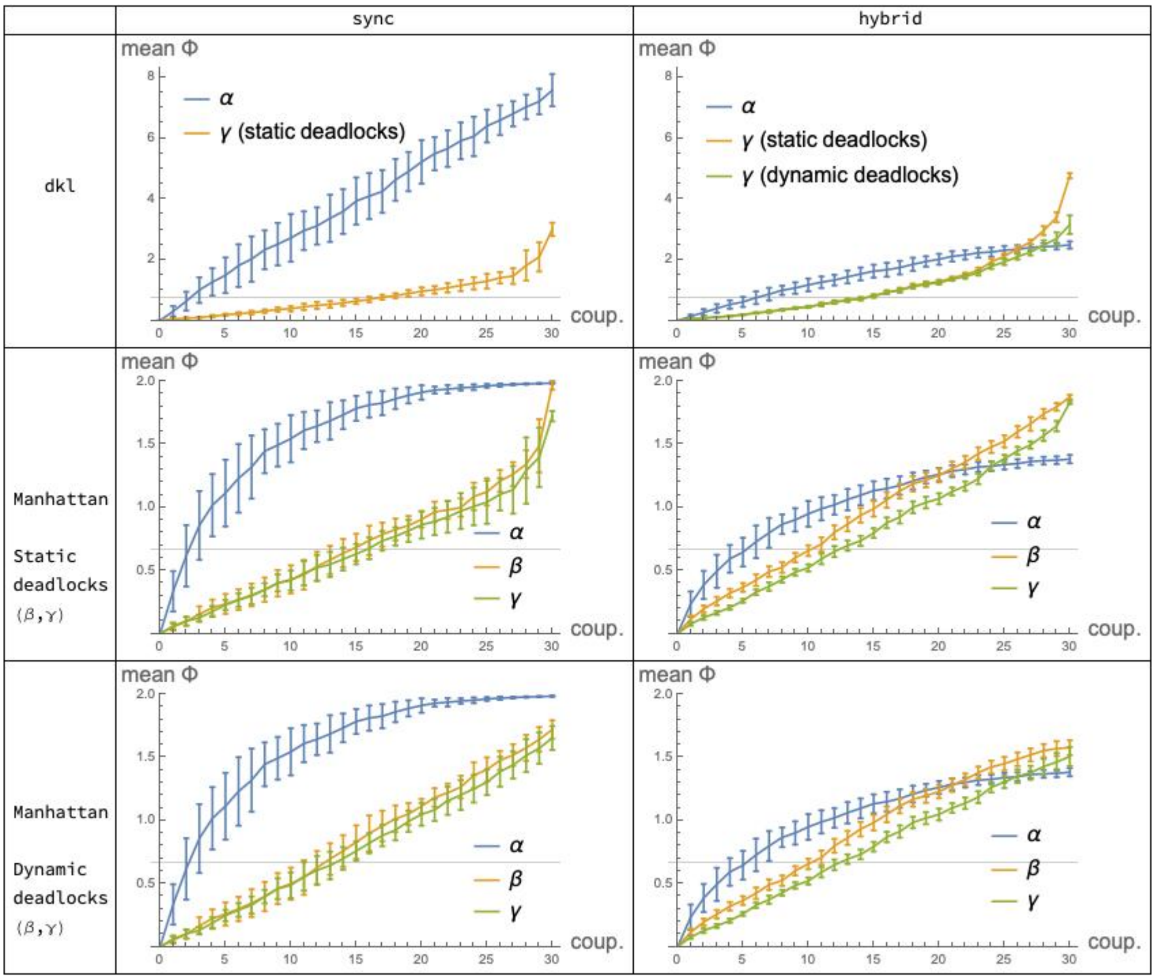

Figure 9. Rearranging the plots of the previous sections for a comparison of the α, β, γ cooperation mechanisms, given a particular choice of execution mode (**columns**) and distance function (**rows**). For Manhattan distance, we differentiate between systems with static or dynamic deadlocks (**Rows 2 and 3**).

In the following subsections, we consider all three "dimensions" (iii), (ii) and (i), precisely in this order, with (i) being the dominant one.

9.1. Distribution Distances: Kullback–Leibler Divergence dkl *vs. Manhattan*

It is not our goal here to assess the various (pseudo-)distances used for defining $\bar{\phi}$: the interested reader can find an accurate study involving *seven* options for this metric in [16]. However, we are interested in checking whether, despite the different ranges of values and plot shapes that they yield, the two alternative distances give analogous indications about the *mutual relations* among the α, β and γ mechanisms.

A quick look at the grid of Figure 9 suggests that the relations between the plots for α and γ are not "qualitatively" different under *dkl* (Row 1) and under Manhattan distance (Rows 2 and 3). By this, we mean that the *relative order* between $\bar{\phi}$ values for the different mechanisms and modes, as the coupling factor moves in its range, is substantially the same for the two distances.

More precisely, we can split the comparisons in two steps.

Fix the mode and vary the mechanism. In mode *sync* (Column 1 of Figure 9), the relation between the α and γ plots is qualitatively the same for *dkl* and for *Manhattan distance*. A similar observation applies relative to mode *hybrid* (Column 2).

Fix the mechanism and vary the mode. Consider the α mechanism: under *dkl*, the relation between $\bar{\phi}^{sync}_{dkl}$ and $\bar{\phi}^{hybrid}_{dkl}$ is depicted in Figure 2; under Manhattan distance, the relation between $\bar{\phi}^{sync}_{manh}$ and $\bar{\phi}^{hybrid}_{manh}$ is qualitatively the same, and is depicted in Figure 3. For mechanism β, the comparison *dkl/Manhattan distance* does not apply. For mechanism γ, under *dkl* the relation between $\bar{\phi}^{sync}_{dkl}$ and $\bar{\phi}^{hybrid}_{dkl}$ is depicted in Figure 7 (left), where static deadlocks are assumed. An analogous relationship is observed between $\bar{\phi}^{sync}_{manh}$ and $\bar{\phi}^{hybrid}_{manh}$ in Figure 8 (left) that also assumes static deadlocks.

In conclusion, the choice to use *Manhattan distance* as an alternative to *dkl*, for comparing the α, β and γ mechanisms appears, a posteriori, convenient and fully legitimate.

9.2. Modes: Sync vs. Hybrid

Having found that the choice of distribution distance does not affect the picture of the relations among α, β and γ, we may wonder whether the same happens with the choice between *sync* and *hybrid* mode. It turns out that this is *not* the case: the pictures that emerge under the two modes are different, as a quick comparison of the two columns of Figure 9 reveals. In light of the findings of the previous subsection, it is sufficient, convenient and safe to focus on Manhattan distance—Tows 2 and 3. Indeed, given the minimal differences between these two rows, choosing one or the other is irrelevant: implementing static or dynamic deadlocks in the *tpm*s does not significantly affect our comparisons.

In *sync* mode, the α plot is constantly higher than the β and γ plots, which are close to each other; in *hybrid* mode, the α plot crosses the other two. This crossing is mainly due to a substantial *decrement* of the values of the α plots, when switching from *sync* to *hybrid* (Figure 3)—a justification of this is given in Section 6.2—whereas for the β mechanism the effect of the mode switch is reversed, with a moderate *increase* of the values of the *hybrid* over the *sync* plot (Figure 5). For the γ mechanisms (Figure 8) the effect of switching from *sync* to *hybrid* is similar.

Why does this happen? Why do the arguments related to distributions spread in Section 6.2 not apply here? Our conjecture is as follows. Under the α mechanism, the much higher abundance of transition possibilities provided by the *hybrid* over the *sync* mode for system $(P{<}\alpha{>}Q)^{hybrid}$ causes a *reduction* of the distance between distributions x_{coop} and x_{indep} in $\phi^{hybrid} = d(x_{coop}, x_{indep})$, as discussed. On the contrary, in system $(P|\beta|Q)^{hybrid}$, written more accurately $P^{hybrid}|\beta|Q^{hybrid}$, the higher abundance of transitions in P^{hybrid} and Q^{hybrid}, with respect to those in P^{sync} and Q^{sync}, offers *more opportunities* to $P^{hybrid}|\beta|Q^{hybrid}$ than to $P^{sync}|\beta|Q^{sync}$ to perform *synchronization transitions* involving P and Q (the reader is invited to recall the discussed difference between *sync* execution mode of a bool net and *synchronization transition* for parallel composition $P|\beta|Q$). More synchronization transitions for $P^{hybrid}|\beta|Q^{hybrid}$ yield a *bigger gap* between distributions $post^{hybrid}_{P|\beta|Q}(X_{PQ})$ and $post^{hybrid}_{P|||Q}(X_{PQ})$ ($P|||Q$ cannot perform any synchronization transition), or between distributions

$pre^{hybrid}_{P|\beta|Q}(Y_{PQ})$ and $pre^{hybrid}_{P|||Q}(Y_{PQ})$ (switching from forward to backward reasoning), the latter being the distributions that feature in the definition of $\phi^{m}_{P|\beta|Q}(Y_{PQ})$ in Equation (19).

The conclusion is that the comparison among α, β and γ cannot be done independently of the execution mode.

9.3. Mechanisms: α vs. β vs γ

We come finally to the comparison among cooperation mechanisms, one that we must contextualize to one or the other execution modes, as just established.

For further assessment of the data in Figure 9, we have included in the plots of Row 1, relative to the *dkl* distance, a gridline corresponding to the expected relative entropy $dkl[rand_1||rand_2]$, where $rand_1$ and $rand_2$ are two *random* distributions. Similarly, in the plots of Rows 2 and 3, relative to Manhattan distance, gridlines indicate the expected Manhattan distance $Manh[rand_1||rand_2]$. These expected values are established by the next two propositions.

Proposition 4. *If* $p = \{p_1 \dots p_N\}$ *and* $q = \{q_1 \dots q_N\}$ *are random discrete distributions over the same domain of size N, where each element* p_i *(respectively,* q_i*) is obtained by picking a real number* x_i *(respectively,* y_i*) at random in (0, 1] and normalizing it so that p (respectively, q) is a probability vector, then as* $N \to \infty$ *the expected KL-divergence between p and q is:*

$$Mean(dkl[p||q]) = 1/ln(4) = 0.72. \tag{32}$$

Proof. We have:

$$\begin{aligned} Mean(dkl[p||q]) &= Mean(\sum_{i=1}^{N}(p_i Log_2(p_i/q_i))) && (33)\\ &= Mean(\sum_{i=1}^{N}(\frac{x_i}{N/2}Log_2(x_i/y_i))) && (34)\\ &= 2Mean(x_i Log_2(x_i/y_i)) && (35)\\ &= 2\int_0^1\int_0^1 xLog_2(x/y)dxdy && (36)\\ &= 2/ln(16) && (37)\\ &= 1/ln(4). && (38) \end{aligned}$$

For Equation (34), we use the definitions of p_i and q_i, and $Mean(\sum_{i=1}^{N} x_i) = N/2$. For Equation (35), we swap *Mean* and *summation* and use the fact that $Mean(x_i Log_2(x_i/y_i))$ is the same for all *i*s. In Equation (36), we express the *Mean* as an integral on the unit square. The integral is routinely solved by parts, yielding Equations (37) and (38), where "*ln*" is the natural logarithm. □

Proposition 5. *If p and q are random distributions of length N as defined in Proposition 4, then as* $N \to \infty$ *the expected Manhattan distance between p and q is:*

$$Mean(Manh[p,q]) = 2/3. \tag{39}$$

Proof. The easy proof is analogous to that of Proposition 4, and is omitted. □

9.3.1. Comparing Mechanisms under the *sync* Mode

The relevant plots for this comparison are those in Column 1 of the grid in Figure 9, Row 2 or 3. $\bar{\phi}$ values for the α mechanism are remarkably higher than those attained by the β and γ mechanisms, which are very close to each other.

Bool nets executing in the *sync* mode exhibit a fully deterministic behavior, and, due to their simplicity, are probably the model most widely used in the literature for illustrating the basic concepts of IIT. If we were to accept them as a sufficiently realistic model for consciousness phenomena, then, according to our findings, we would conclude that the traditional, simple α cooperation mechanism, outperforms the alternative and, in a way, more sophisticated process-algebraic mechanisms β and γ in achieving high values of integrated information and, potentially, consciousness. This gap is particularly marked in the central segment of the range of coupling values, where P and Q show an even balance of cooperation and independence. However, the ever-lasting debate on determinism vs. nondeterminism in the natural sciences must invest also the neurosciences, and, although we cannot provide an accurate picture of the status of this discussion in this field, we believe that the assumption of a fully deterministic model for the brain appears too restrictive, if not naive. The *hybrid* mode may then be a better option. (Of course, the nondeterministic *hybrid* mode appears a better option *if* we restrict ourselves to the relatively small family of models investigated in the paper, but there may well be other nondeterministic models that lend themselves to interesting and perhaps more appropriate and realistic applications to neuroscience. For example, an option that seems to have gained attention inside the IIT community (as emerged in a private communication) is that of noisy mechanisms run in *sync* mode, e.g., the idea that the computations performed by the boolean functions associated with bool net nodes are affected by some percentage of error.)

9.3.2. Comparing Mechanisms under the *Hybrid* Mode

The *hybrid* mode produces nondeterministic behavior. Nondeterminism *may* appear as a desirable feature, when dealing with complex systems and models in neuroscience; however, it is certainly a *must* for system models in Software Engineering. In the early phases of software development, for example, nondeterminism is typically used to prevent premature design choices that are postponed to later phases, down to the final implementation—a convenient way to offer implementation freedom. Then, if we set up to assess the three cooperation mechanisms in the context of formal models for Software Engineering, or system engineering in general, we believe that it is much more appropriate to refer to the *hybrid* execution mode.

Here, the relevant plots are those in the last column of Figure 9, Row 2 or 3. The $\bar{\phi}$ plots for the LOTOS-inspired mechanisms β and the CCS-inspired mechanism γ end up performing in a similar way, as it happens under the *sync* mode. However, the difference between α, on the one hand, and β-γ, on the other hand, is now considerably reduced, and a faster growth of the α plot in the lower part of the *coup* range is counterbalanced by the slightly higher values of the β-γ plots in the upper part.

It seems arduous, and perhaps even pointless, to speculate on the *detailed* differences among those three plots, trying to justify them formally—detailed differences that one may well expect, given the substantial difference between the *sharVar* and the *sharTrans* mechanisms. On the other hand, by taking a *coarser* look at the mentioned plots, we can reasonably conclude that, in terms of integrated information, *the performances of the three mechanisms are roughly equivalent.* Furthermore, by comparing their plots with the reference values (the gridlines) that derive from purely random distributions, we can additionally claim that all three methods of coupling two systems P and Q for them to interact, *do their jobs quite well*: in their highest values, all of them roughly double those reference values.

10. Conclusions

In this paper, we have addressed the scenario of two state transition systems P and Q that exhibit different types of cooperation and a variable degree of coupling. We have applied the informational measure of averaged *Integrated Information* $\bar{\phi}$ for the assessment and comparison of two fundamentally different cooperation mechanisms: (i) the standard *shared variables* mechanism associated to bool nets and very often adopted in the IIT literature, expressed as $P{<}\alpha{>}Q$; and (ii) the *shared transitions* mechanism typical of process algebras, which we have studied in the two forms $P|\beta|Q$ and $P[\gamma]Q$. In each case, α, β and γ control and measure the degree of coupling between P and Q. Having been able to export $\bar{\phi}$ from its standard application context and to adapt it to a completely novel field, re-defining what *cooperation* and *independence* mean in the new setting, is, in our opinion, one of the interesting and original contributions of our work, on the conceptual side.

We have modeled P and Q as boolean nets, and have considered three possible *execution modes*, namely *synchronous*, *asynchronous* and *hybrid*, although the anomalies of the second mode soon suggested to drop it. Furthermore, we have considered two variants of Integrated Information, based on two distinct measures of *distribution distance*, namely Kullback–Leibler divergence "*dkl*" and Manhattan distance, which avoids some limitations of dkl. With the main objective to compare the α, β and γ cooperation mechanisms (*coop*), the idea to articulate our experimental analysis along those two additional dimensions—execution modes (m) and distribution distances (dd)—has been useful for obtaining a sufficiently large set of plots for

$$\bar{\phi}^{m}_{dd}(\text{P-}coop\text{-Q})$$

on which to ponder.

In summary, the inspection of these plots has led us to the following main conclusions.

- Adopting a definition of $\bar{\phi}$ based on Manhattan distance rather than dkl makes averaged integrated information more widely applicable; furthermore, when both variants apply, they yield nicely compatible indications. For our purposes, Manhattan distance is therefore more convenient, and safe. It is worth noting that the Earth Mover's Distance (EMD) [17], adopted in IIT 3.0 [2] but computationally more costly that Manhattan distance, would also avoid the mismatch problem arising with dkl.
- Under the deterministic, *sync* execution mode, the IIT-standard cooperation mechanism α performs considerably better than β and γ, especially for bipartite systems structured so that the two parts exhibit an intermediate degree of coupling. Conversely, under the nondeterministic *hybrid* mode, which may be more appropriate for cognitive system models, but is definitely more appropriate for Software Engineering models, the three mechanisms exhibit roughly equivalent performances, and *good* ones, at least compared with those achieved by using randomized state distributions.

Could the latter approximate equivalence be intuitively expected a priori? For the author, this was not the case. Let us explain.

In the general context of discrete state-transition models for distributed, concurrent systems, it seems reasonable to consider a cooperation mechanism as *effective* when the cooperating parts can produce state distributions—*successor* or *predecessor* state distributions, corresponding to *effect*- or *cause*-reasoning—that are markedly different from those achieved when the parts work independently. The reason one might expect, at least for the *hybrid* mode, higher $\bar{\phi}$ values for the *sharTrans* than for the *sharVar* mechanism has to do with the difference in *intrinsic complexity* of the two mechanisms.

For simplicity, let us refer to effect reasoning, i.e., successor state distributions. For finding the next state y_{PQ} under α-cooperation, we only need to evaluate the n boolean functions of the bool net, where n is the overall number of nodes; the interactivity between P and Q comes "for free", depending only on the fact that, for intermediate values of α, both P and Q read a mix of local and remote nodes.

Under β- and γ-cooperation, boolean function evaluation is still necessary, but then the possible local transition labels must be computed, and these depend both on current local states x_P and x_Q and on next states y_P and y_Q, according to our labeling policy. Then, transition $x_{PQ} \rightarrow y_{PQ}$ is determined after a sort of negotiation between P and Q, based both on the locally available labeled transitions and on the set β of synchronization labels. It is clear that mechanisms β and γ are intrinsically more complex than α: they manipulate more information and do more work. It seemed then plausible that these mechanisms were able to exploit this additional information and machinery for creating next state distributions y_{coop} that can depart more markedly from the reference distribution y_{indep}, thus achieving higher $\bar{\phi}$ values. (We also expected that the noise injected in $P*$

and $Q*$ for computing the distribution product $pre^m_{P*}(Y_P) \times pre^m_{Q*}(Y_Q)$ in Equation (15) could act as a limiting factor for the gap between this distribution and distribution $pre^m_{P<\alpha>Q}(Y_{PQ})$, thus limiting the growth of $\bar{\varphi}^{hybrid}(P{<}\alpha{>}Q)$, and keeping it well below $\bar{\varphi}^{hybrid}(P|\beta|Q)$ and $\bar{\varphi}^{hybrid}(P[\gamma]Q)$.)

Experimental evidence has shown that this expectation was wrong.

The combination of measure $\bar{\varphi}$ with mechanisms β and γ may appear bizarre to the IIT expert ("why β and γ") as well as to the expert in process algebra ("why $\bar{\varphi}$").

The first expert may criticize the adoption of additional interaction mechanisms for modeling brain-like systems, when several phenomena related to consciousness have already been successfully investigated by the plain bool net model without super-imposed features, and given that parallel compositions $|\beta|$ and $[\gamma]$ fail to satisfy conditional independence. Nevertheless, flat bool nets using the basic α mechanism suffer from serious *scalability* problems. The state space of an (n, k)-bool net has size 2^n and the associated tpm is a $2^n \times 2^n$ matrix: given this exponential growth, standard computers and algorithms can successfully deal only with "toy" models (such as those investigated in this paper), while there is no hope to handle realistic systems whose size n is, e.g., 5 or 10 orders of magnitude larger. We still believe that exploring macro-structured bool nets and higher-level interaction mechanisms—β, γ or others—may help alleviate those problems.

The process-algebra expert, in turn, may be puzzled by the fact that $\bar{\varphi}$ is defined in terms of just *one-step* $x_{PQ} \rightarrow y_{PQ}$ of system behavior, but considers the full repertoire of conceivable system states, *including those unreachable from the initial state* (in this respect, $\bar{\varphi}$, especially in the state-independent form, reflects the "counterfactual-reasoning" that informs J. Pearl's Do-Calculus of intervention [9]) $\bar{\varphi}$ may then appear: (i) inadequate to cope with systems in which the existence and importance of an initial state is out of question—as in Process Algebra and Software Engineering; and (ii) insensitive to phenomena that emerge only with longer transition sequences, e.g., attractors (the interested reader may look at the demonstration in [18], where attractors play a role for the analysis of *asymptotic* mutual information between boolean net components P and Q). With respect to this objection, we do agree that the two analytical approaches of *one-step-from-all-states* and *all-steps-from-one-state* may address and reveal different system properties, but, of course, we can take them as *complementary* techniques and explore their potential synergy. In any case, using $\bar{\varphi}$ in the context of formal models and languages for software engineering is, to our knowledge, a radically novel way to assess the "power" of their structuring principles and operators.

Funding: This research was partially funded by FQXi, Mini-Grant number FQXi-MGA-1702.

Acknowledgments: I express my gratitude to Larissa Albantakis for many inspiring exchanges, both by e-mail and in person. Many thanks also to my colleagues Paola Favati, for the manual solution of the integration step in the proof of Proposition 4, and Franco Mazzanti, for illuminating discussions on the early versions of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Balduzzi, D.; Tononi, G. Integrated Information in Discrete Dynamical Systems: Motivation and Theoretical Framework. *PLoS Comput. Biol.* **2008**, *4*, e1000091. [CrossRef] [PubMed]
2. Oizumi, M.; Albantakis, L.; Tononi, G. From the Phenomenology to the Mechanisms of Consciousness: Integrated Information Theory 3.0. *PLoS Comput. Biol.* **2014**, *10*, e1003588. [CrossRef] [PubMed]
3. Kauffman, S.A. Homeostasis and Differentiation in Random Genetic Control Networks. *Nature* **1969**, *224*, 177–178. [CrossRef] [PubMed]
4. Nicola, R.D. Process Algebras. In *Encyclopedia of Parallel Computing*; Padua, D.A., Ed.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 1624–1636.
5. Milner, R. *A Calculus of Communicating Systems*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1980; Volume 92.
6. Hoare, C.A.R. *Communicating Sequential Processes*; Prentice Hall: Upper Saddle River, NJ, USA, 1985.

7. Bergstra, J.; Klop, J.W. Process Algebra for Synchronous Communication. *Inf. Control* **1984**, *60*, 109–137. [CrossRef]
8. Bolognesi, T.; Brinksma, E. Introduction to the ISO Specification Language LOTOS. *Comput. Netw. ISDN Syst.* **1987**, *14*, 25–59. [CrossRef]
9. Pearl, J. *Causality: Models, Reasoning and Inference*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2009.
10. Cover, T.M.; Thomas, J.A. *Elements of Information Theory*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2006.
11. Plotkin, G.D. A Structural Approach to Operational Semantics. *J. Log. Algebr. Program* **2004**, *60–61*, 17–139.
12. Brinksma, E. Constraint-Oriented Specification in a Constructive Formal Description Technique. In *Stepwise Refinement of Distributed Systems, Models, Formalisms, Correctness, Proceedings of the REX Workshop, Mook, The Netherlands, 29 May–2 June 1989*; de Bakker, J.W., de Roever, W.P., Rozenberg, G., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1989; Volume 430, pp. 130–152.
13. Bolognesi, T.; Derrick, J. Constraint-oriented Style for Object-Oriented Formal specification. *IEE Proc. Softw.* **1998**, *145*, 61–69. [CrossRef]
14. Bolognesi, T. Integrated Information in Partitioned Boolean Nets. The Wolfram Demonstrations Project. 2019. Available online: http://demonstrations.wolfram.com/IntegratedInformationInPartitionedBooleanNets/ (accessed on 16 August 2019).
15. Oizumi, M.; Amari, S.; Yanagawa, T.; Fujii, N.; Tsuchiya, N. Measuring integrated information from the decoding perspective. *CoRR* **2015**. [CrossRef] [PubMed]
16. Tegmark, M. Improved Measures of Integrated Information. *PLoS Comput. Biol.* **2016**, *12*. [CrossRef] [PubMed]
17. Rubner, Y.; Tomasi, C.; Guibas, L.J. The Earth Mover's Distance as a Metric for Image Retrieval. *IJCV Int. J. Comput. Vis.* **2000**, *40*, 99–121. [CrossRef]
18. Bolognesi, T. Mutual Information between Boolean Net Regions. The Wolfram Demonstrations Project. 2018. Available online: https://demonstrations.wolfram.com/MutualInformationBetweenBooleanNetRegions/ (accessed on 16 August 2019).

© 2019 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).