Homology, Homotopy and Applications, vol.5(2), 2003, pp.95–136

SOME GEOMETRIC PERSPECTIVES IN CONCURRENCY THEORY

ERIC GOUBAULT

(*communicated by Gunnar Carlsson*)

Abstract

Concurrency, i.e., the domain in computer science which deals with parallel (asynchronous) computations, has very strong links with algebraic topology; this is what we are developing in this paper, giving a survey of "geometric" models for concurrency. We show that the properties we want to prove on concurrent systems are stable under some form of deformation, which is almost homotopy. In fact, as the "direction" of time matters, we have to allow deformation only as long as we do not reverse the direction of time. This calls for a new homotopy theory: "directed" or di-homotopy. We develop some of the geometric intuition behind this theory and give some hints about the algebraic objects one can associate with it (in particular homology groups). For some historic as well as for some deeper reasons, the theory is at a stage where there is a nice blend between cubical, ω-categorical and topological techniques.

1. Introduction

Concurrency theory deals with systems in which several computational activities (called processes in general) can be performed at the same time, in an asynchronous manner. These were introduced in order to have increased computational power, so that computations can be faster (essentially in scientific computing), or so that some concurrent transactions can be handled efficiently (user interfaces, embedded systems reacting to the external environment etc.) or just handled at all (mostly because of the amount of memory needed, as for concurrent databases).

The variety of applications that motivated the use of concurrent machines has led to many different architectures. The main problem in concurrency is to have processes cooperating for a common goal. Cooperation implies some form of synchronisation and information passing. This can be done through message passing for instance. In this class of models, processes have their own local memory, which cannot be accessed by other processes. The way to communicate values to other processes is by explicitely sending values to these other processes, which will have

Received October 11, 2001, revised July 23, 2002; published on April 22, 2003.
2000 Mathematics Subject Classification: 18F20,54F05,55P15,55U10,68Q55,68Q60,68Q85
Key words and phrases: Homology, Homotopy, Concurrency, Cubical Sets, Di-homotopy

to explicitely ask for receiving values. One of the first of this class of models, is the rendez-vous model (as used in most process algebra, like CCS [**51**], CSP [**37**] etc.) in which the action of sending is blocking the sender until the receiver actually receives the corresponding message. Symmetrically the action of receiving blocks the receiver until the message is actually sent. This is the simplest of all message-passing models (also called synchronous message-passing). The variations of it include, non-blocking send but blocking receive, non-blocking send and receive (asynchronous message-passing), broadcasts to groups of processes instead of "point to point" communication etc.

Another important class of concurrent architectures is shared memory style. Here, processes have a local memory indeed, where they can perform their local computations, but also have a common memory space, which is accessible to all. Communication between processes is essentially asynchronous and is realized by writing and reading values in this common space, as pictured in Figure 1. Processes $P_1, \cdots, P_n$ are writing and reading through shared locations such as scalar variables x and z (containing boolean or integer values for instance), and also through more complex structures, such as y, a "3-cell buffer" i.e., a variable consisting of a queue of 3 values. If concurrent reading by several processes is not a problem in general, concurrent writing of scalar variables is not to be allowed. At the hardware level, this would mean at best, undefined behaviour, and at worst, short circuit. Therefore, it is necessary to "protect" the accesses to shared variables by some mechanism. A classical one is by using "semaphores" introduced by E. W. Dijkstra [**10**] in 1968.

Basically, before a process tries to write on a location in a shared memory, it has to put a "lock" on it (through its associated semaphore), blocking the other processes which try to write at the same time and on the same location. Formally, the action of putting a lock on location x is denoted by Px (using E. W. Dijkstra's notation [**10**]). In case x is some more complex structure than a read/write variable (such as y above), at most $n \geqslant 1$ processes can hold a lock on x (here with z, $n = 3$) before blocking the accesses by other processes. In this case we call the associated semaphore an n-semaphore[1]. After some process has written what it needed to write on x, it can safely relinquish its lock by doing action Vx; this will allow another process to acquire a lock on x, i.e., to allow it to resume its execution.

Let us forget about actual calculations on x, y, z etc. and focus only on the locking, unlocking mechanism (the coordination of processes involved). We will then identify shared locations with their associated n-semaphores. This urges us to consider throughout this paper (except for some minor exceptions) a simplified programming language, in which processes are regular expressions on the alphabet $\{Pa, Va \mid a \in Loc\}$, where Loc is a set of "locations". Each of these locations are in fact n-semaphores, for some n, defined by a map $s : Loc \to \mathbb{N}$. Regular expressions are formed freely from the alphabet $\{Pa, Va \mid a \in Loc\}$ by application of the

[1] In fact, a semaphore has an associated integer which counts the number of processes which can still lock it; each lock decreases the counter, each unlock increases it. Processes trying to lock a zero-valued semaphore have to wait for another process to relinquish a lock. When the semaphore is initialized with value $n > 1$, it can be locked by at most n processes concurrently; it is called a counting semaphore in operating systems theory. When it is initialized with value one, it is called a binary semaphore. We use the term n-semaphore in the two cases for the sake of simplicity.

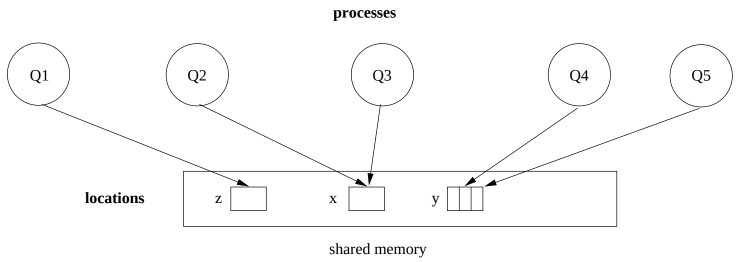


Figure 1: A shared memory concurrent machine.

following algebraic operators: + (which is associative and commutative), . (which is associative), and the unary operator *. "Elementary moves" (or actions) are elements of the alphabet, i.e., Pa, Vb etc. $A + B$ means that sequences of actions that can be taken are those of A or those of B – this is non-deterministic choice. Sequences of actions of $A.B$ are concatenations of sequences of actions of A and of actions of B (this is the concatenation operation), and sequences of actions of A^* are any number of concatenations of sequences of actions of A (this is the Kleene star operation, or finite unbounded iteration).

What are we looking for now? We want to be able to derive properties of concurrent machines, even of such a simplified one. Of course, the theory of sequential computation is very much advanced and the properties of interest for sequential computation (what function of the arguments are we computing? Is the computation always terminating for all its arguments? How long will this take? etc.) are not the ones we are dealing with here. The novelty in concurrent programming resides not in the fact we are computing another class of functions (which would contradict Turing's thesis) but is the fact that coordination between processes does matter. For instance, we might have forgotten to properly lock some locations, creating an unexpected behaviour of the program. On the contrary we might have constrained the coordination too much, preventing the program to carry out normal computation. This is called a deadlocking situation. Another property of interest is to know whether a concurrent system can go into a "bad state" or not. Typically, we are trying to solve a "reachability problem", e.g., do we have an execution in our system which will go through such bad states? Also, we can ask for slightly more subtle properties: for some applications (we will see an example later on), some sequences of accesses to resources are considered right while others are not. It is therefore of primary importance to be able to classify such sequences; this will actually lead to arguments using homotopy theory.

Before getting to this, let us briefly show how this would normally be dealt with. Of course to be able to prove things, one needs a mathematical model, in particular for the notion of execution (sequence of actions) in a concurrent system.

There is a great variety of models for concurrency, as witnessed in [**68**] for instance. Transition systems are one of the oldest semantic models, both for sequential

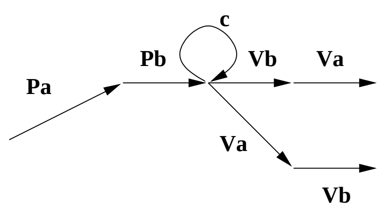


Figure 2: A simple (sequential) transition system.

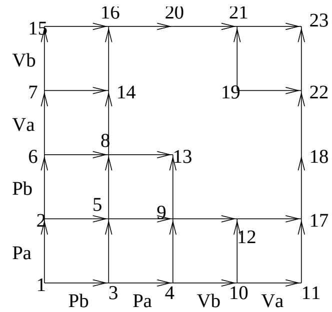


Figure 3: A transition system interpreting $Pb.Pa.Vb.Va \mid Pa.Pb.Va.Vb$.

and concurrent systems:

Definition 1. *A transition system is a structure (S,i,L,Tran) where,*

- *S is a set of states with initial state i*
- *L is a set of labels, and*
- *$Tran \subseteq S \times L \times S$ is the transition relation*

Transitions systems are nothing but discrete dynamical systems: in general the transition relation $Tran$ is represented as a directed graph of actions. For instance the transition system depicted in Figure 2 gives semantics to the process $Pa.Pb.c^*.(Va.Vb + Vb.Va)$, i.e., to a process which locks a, then b then does some sequence c any finite number of times (this can be a computation on a and b), then unlocks a and b in any order. This behaviour can be seen by looking at paths (or executions) in this directed graph, from the leftmost state (the initial state) to the rightmost ones (the final states).

A simple way to look at processes in parallel is to build a transition system for each process and then to construct some kind of fibered product of all these graphs of actions (this has a formal sense, see for instance [**1**]): states of this transition system are now tuples of states of each individual process, and transitions from one to another are interleavings of transitions of each individual process. For instance, the graph of actions for $T_1 = Pb.Pa.Vb.Va$ in parallel with $T_2 = Pa.Pb.Va.Vb$ is shown in Figure 3. State 1 is the initial state, actions on parallel segments have the same label.

Now we can see that state 13 is not a "correct" final state. State 23 consists of the pair of endpoints of digraphs representing each process, but not 13, which has nevertheless no future. This is a deadlock. In this situation, the first process T_1 has a lock on b, waiting for a lock on a where the second one T_2 has a lock on a waiting for a lock on b. This is typical of a "deadly embrace" as E. W. Dijkstra originally put it.

We can also ask ourself whether this concurrent system can be in a state we do not want (which is rather artificial here); this would be a state in which T_1 would have a lock on a and just released a lock on b, whether T_2 would have a lock on b

and just released a lock on a. Looking at the graph of Figure 3 one sees that this is precisely state 19, but there is not path from the initial state 1 to 19, so we never go through this "bad state".

Last but not least, we can also try to classify the different access orders to resources in this system. Looking at all the 10 paths from state 1 to state 23 in the directed graph of Figure 3, we see that we have only two such orders: T_1 holds locks on b then a before T_2 does, or T_2 holds locks on a then b before T_1 does. This situation is typical of concurrent databases, and is known under the name "serializability".

A distributed database can be seen as a shared-memory machine (containing items) on which processes (called transactions) act by reading and writing, getting permissions to do so by using the appropriate functions on attached semaphores. One of the main purposes of this area is to ensure coherence of the distributed database while ensuring good performance, through a definition of suitable policies (protocols) for transactions to perform their own actions (with P and V). This entails of course that deadlock-freedom of transactions is of importance. Correctness of a distributed database is itself very often expressed by some form of a *serializability* condition.

Suppose we have two transactions $T_1 = Pb.Vb.Pa.Va$ and $T_2 = Pa.Va.Pb.Vb$ trying to modify two items a and b. There is a path of execution in which T_1 acquires b, T_2 acquires a, then T_1 acquires a and T_2 acquires b. Think of the database to represent airplane tickets (for instance b is the return ticket corresponding to the one-way ticket a), and the two transactions to represent remote booking booths, the action between a P and its corresponding V is writing a name on the ticket. The situation here is that T_1 will have reserved its one-way ticket and T_2 will have reserved its return ticket only. This is not an allowed behaviour. It is not equivalent to a purely serial schedule which are the only ones that are specified as correct (only one of T_1 or T_2 gets the whole lot of tickets). Of course, this could be seen on the corresponding transition system, but if we have many complex processes running altogether, the "state-space" and therefore the path-space becomes enormous. Therefore it is important to have a way to "retract" this onto smaller transition systems (or shapes) where we can still observe similar state or path like properties. This is where some ideas from algebraic topology sneak in. We will see in the next sections how this can be made precise.

Organisation of the paper. In Sections 2, 3, 4 and 5, we show how to model these phenomena using, in a natural way, concepts from topology and combinatorial algebraic topology. This will give us a meaning for the terms used above, such as "retract", first in a topological model (Section 2) and then in a combinatorial model (Sections 3 and 4). This is all based on a notion of deformation, or homotopy, which is slightly different from the usual homotopy of topological spaces. Here the direction of time should be preserved, meaning the maps and hence the homotopies considered should preserve the time direction. This is why the newly defined homotopy theory is called directed homotopy or "dihomotopy".

To fully reflect the combinatorial model, we have to refine the topological model of Section 2; this is done in Section 5. Then we can attack in Section 6 a first geometric study of the notions such as deadlocks, schedules, serializability conditions etc. This is only a first step, ideally one should try to find computable invariants of dihomotopy. Some leads are given in Section 7, but there again, this implies some refinement of the modeling, to have nice and "precise" functors; some of which are shown in Section 8. Then one can try to see if standard results, such as Seifert/van Kampen or exact sequence theorems still hold in the new theory. Some hints are given in Section 9. We conclude by some perspectives in Section 10.

Some further references. The "topological" formalization that follows is one of the most recent ones, and essentially dates back to [**14**] and [**15**], but is based on much older results [**10**].

The combinatorial (cubical) and homological calculations are older, and have been at the center of [**26**], starting with [**30**], [**25**] and [**27**].

I actually only realized the relationships between the combinatorial and the topological approaches quite recently, and have been aware of this line of research only after J. Gunawardena published his very enlightening paper [**35**].

There are some ideas about using n-categories in [**54**]. It is only quite recently that these have come into full bloom, see [**21**] for a start, where many algebraic invariants are also introduced. The "unification" of these approaches has lead us to the concept of a globular CW-complex [**23**] which I will briefly describe in Section 8.

The interested reader can find more references about this in [**28**] or [**15**].

2. A topological approach

The first "algebraic topological" model I am aware of is called a *progress graph* and has appeared in operating systems theory, in particular for describing the problem of "deadly embrace" in "multiprogramming systems". *Progress graphs* are introduced in [**9**], but attributed there to E. W. Dijkstra. In fact they also appeared slightly earlier (for editorial reasons it seems) in [**60**].

The basic idea is to give a description of what can happen when several processes are modifying shared resources. Given a shared resource a, we see it as its associated semaphore that rules its behaviour with respect to processes. For instance, if a is an ordinary shared variable, it is customary to use its semaphore to ensure that only one process at a time can write on it (this is mutual exclusion). Then, given n deterministic sequential processes $Q_1, \ldots, Q_n$, abstracted as a sequence of locks and unlocks on shared objects, $Q_i = R^1 a_i^1 . R^2 a_i^2 \cdots R^{n_i} a_i^{n_i}$ (R^k being P or V for respectively acquiring and releasing a lock on a semaphore), there is a natural way of understanding the possible behaviours of their concurrent execution, by associating to each process a coordinate line in $\mathbb{R}^n$. The state of the system corresponds to a point in $\mathbb{R}^n$, whose ith coordinate describes the state (or "local time") of the ith processor.

Consider a system with finitely many processes running concurrently. We assume that each process starts at (local time) 0 and finishes at (local time) 1; the P and V actions correspond to sequences of real numbers between 0 and 1, which reflect the order of the P's and V's. The initial state is $(0, \ldots, 0)$ and the final state

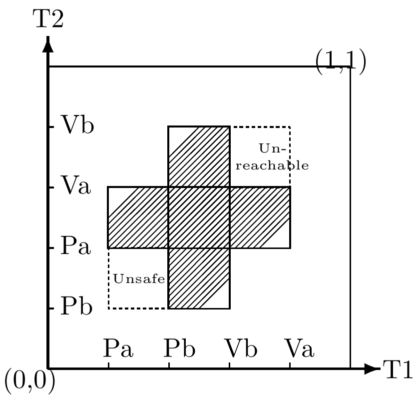


Figure 4: Example of a progress graph

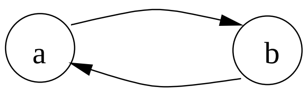


Figure 5: The corresponding request graph

is $(1, \ldots, 1)$. An example consisting of the two processes $T_1 = Pa.Pb.Vb.Va$ and $T_2 = Pb.Pa.Va.Vb$ gives rise to the two dimensional *progress graph* of Figure 4.

The shaded area represents states which are not allowed in any execution path, since they correspond to mutual exclusion. Such states constitute the *forbidden area*. For instance, look at Figure 4 again and take a point whose abscissa is (strictly) between local times marked as Pb and Vb and whose ordinate is (strictly) between local times marked also as Pb and Vb. Having these coordinates means that both T_1 and T_2 have acquired b and not relinquished it, which is impossible since b can only be held by at most one process at a time. This point ought to be forbidden.

An *execution path* is a path from the initial state $(0, \ldots, 0)$ to the final state $(1, \ldots, 1)$ avoiding the forbidden area and increasing in each coordinate - time cannot run backwards. We call these paths *directed paths* or dipaths. This entails that paths reaching the states in the dashed square underneath the forbidden region, marked "unsafe" are deemed to deadlock, i.e., they cannot possibly reach the allowed terminal state which is $(1, 1)$ here. Similarly, by reversing the direction of time, the states in the square above the forbidden region, marked "unreachable", cannot be reached from the initial state, which is $(0, 0)$ here. Also notice that all terminating paths above the forbidden region are "equivalent" in some sense, given that they are all characterized by the fact that T_2 gets a and b before T_1 (as far as resources are concerned, we call this a *schedule*). Similarly, all paths below the forbidden region are characterized by the fact that T_1 gets a and b before T_2 does.

On this picture, one can already recognize many ingredients that are at the center of the main problem of algebraic topology, namely the classification of shapes modulo "elastic deformation". As a matter of fact, the actual coordinates that are chosen for representing the times at which Ps and Vs occur are unimportant, and these can be "stretched" in any manner, so the properties (deadlocks, schedules etc.)

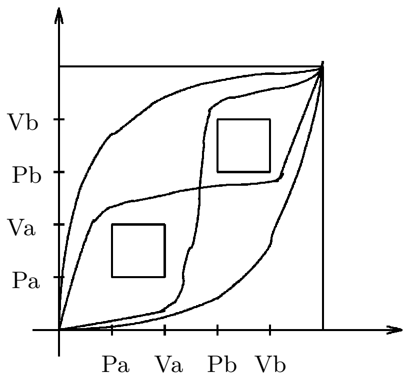


Figure 6: The progress graph corresponding to $Pa.Va.Pb.Vb \mid Pa.Va.Pb.Vb$

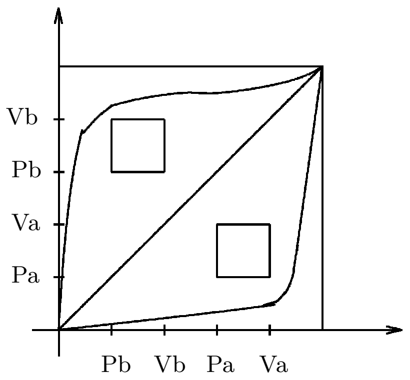


Figure 7: The progress graph corresponding to $Pb.Vb.Pa.Va \mid Pa.Va.Pb.Vb$

are invariant under some notion of deformation, or *homotopy*. This is a particular kind of homotopy though, and this will be at the center of many difficulties in later work. We call it (in subsequent work) *directed homotopy* or *dihomotopy* in the sense that it should preserve the direction of time. For instance, the two homotopic shapes, both of which have two holes, of Figure 6 and Figure 7 have a different number of dihomotopy classes of dipaths. In Figure 6 there are essentially four dipaths up to dihomotopy (i.e., four schedules corresponding to all possibilities of accesses of resources a and b) whereas in Figure 7, there are essentially three dipaths up to dihomotopy.

Before going to the formalization, we should ask ourselves if there is not a simpler way to model these properties.

There is another method to determine deadlocks in such situations, which was of course known long ago and was entirely graph-theoretic, known as the *request graph*. Figure 5 depicts the request graph corresponding to the progress graph of Figure 4. Nodes of this graph are resources of the concurrent system, i.e., here, semaphores. There is an directed edge from a resource x to a resource y if there is a process which has locked x and needs to lock y at a given time. A sufficient condition for such systems to be deadlock-free is that their corresponding request graphs be acyclic[2]. Unfortunately, this is not a necessary condition in general. For instance a request graph cannot capture the notion of n-semaphore with $n \geqslant 2$, i.e., resources that can be shared by up to n processes but not $n+1$ (for instance, asynchronous buffers of communication of size n which can be "written" on by at most n processes). This really calls for some higher-dimensional versions of graphs.

There is no need in fact to resort to fancy n-semaphores to see that request graphs are not enough. Consider the following concurrent program, which is composed of processes `A`, `B` and `C` in parallel (introduced for other reasons by Lipsky and Pa-

[2]Note that this is a very geometric condition indeed.

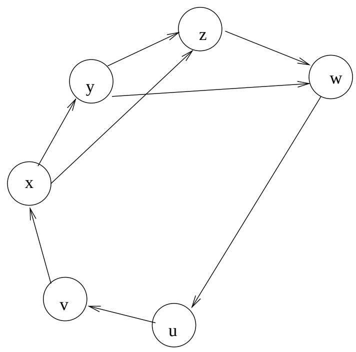


Figure 8: The request graph for the Lipsky/Papadimitriou example.

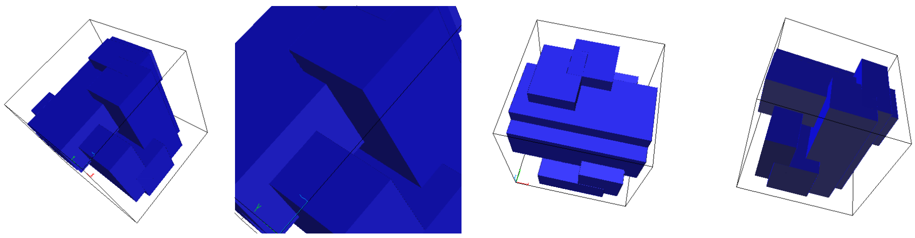


Figure 9: Progress graph of the Lipsky/Papadimitriou example.

padimitriou, mentioned in [**35**]), its request graph (Figure 8) and the corresponding progress graph[3] viewed from different angles in Figure 9:

```
A=Px.Py.Pz.Vx.Pw.Vz.Vy.Vw
B=Pu.Pv.Px.Vu.Pz.Vv.Vx.Vz
C=Py.Pw.Vy.Pu.Vw.Pv.Vu.Vv
```

The request graph for this example contains cycles, but it can be proved that it does not deadlock.

A progress graph can be seen as a topological space - a sub-space of $\mathbb{R}^n$ in fact. The topology is necessary to formally define the notion of path, which has to be continuous (executions cannot skip from one point to another in no time). We also need a partial order, allowing to characterize the "direction" of the time flow, i.e., to characterize future and past of points (which are states of the concurrent system). The two should be at least minimally compatible. We should be able to take (topological) limits under the partial order sign, leading to the following definition:

Definition 2. *[**15**] A po-space (or partially ordered space) is a pair $(X, \leqslant)$ formed by a topological space X together with a partial order $\leqslant$ such that $\leqslant$ is closed (i.e.,*

[3]The holes in the cube of states are in fact represented as solid shapes.

$\leqslant$ is a closed subset of $X \times X$ with the product topology).

This implies two natural properties: the sets $\uparrow x = \{y \mid x \leqslant y\}$ and $\downarrow x = \{y \mid y \leqslant x\}$ are closed subsets of X.

We need now to define suitable morphisms between po-spaces, which in turn will give the notion of dipath.

Definition 3. *Let $(X, \leqslant_X)$ and $(Y, \leqslant_Y)$ be two po-spaces.*

A continuous function $f : X \to Y$ is called a dimap *if for all $y, z \in X$: $y \leqslant_X z \Rightarrow f(y) \leqslant_Y f(z)$.*

A dipath on X is then a dimap $f : \vec{I} \to X$ where $\vec{I}$ is the topological space $I = [0, 1] \subseteq \mathbb{R}$ with the (global) partial order inherited from the one of $\mathbb{R}$. We write $\vec{P}_1(X)$ for the set of dipaths on X and $\vec{P}_1^{\alpha,\beta}(X)$ for the set of dipaths on X going from α to β.

Now, we can define more precisely what we mean by deformation of dipaths, which we call directed homotopy, or dihomotopy. It is very important here to fix the extremities of dipaths. The idea is that, contrarily to the classical case where it suffices[4], in order to characterize a shape, to choose a basepoint and then to consider loops around this basepoint modulo homotopy, here we really need two basepoints[5]. As a matter of fact, it is very unlikely that we have lots of directed loops in our shapes (in fact, there are only trivial constant directed loops in a progress graph) so we have to choose a source basepoint and a target basepoint, and then study all dipaths between these two points modulo dihomotopy.

Definition 4. *[**15**] Let f and g be two dipaths on X between an initial point α and a final point β. A dihomotopy between f and g is a dimap from $\vec{I} \times I$ to X such that for all $x \in \vec{I}$, $t \in I$, $H(x, 0) = f(x)$, $H(x, 1) = g(x)$ and $H(0, t) = \alpha$, $H(1, t) = \beta$. We write $f \sim g$.*

A first example of the kind of properties one wants to check on some systems, which involves a characterization of dihomotopy classes of dipaths is the "serialization" property in some concurrent databases.

Look for instance at Figure 7 which we have already mentioned in the introduction. All paths of execution above the left hole are equivalent to a serial execution of transaction T_2 then transaction T_1. All paths of execution below the right hole are equivalent to the serial execution of transaction T_1 then T_2. The third type of dipath is not a serial dipath: it describes several equivalent cases, for instance: T_1 acquires b, T_2 acquires a, then T_1 acquires a and T_2 acquires b.

Can we now make sense of the serializability condition of the introduction? A simple criterion can seem to do the job here. It seems on the example of Figure 7 that connectivity of the forbidden region is a necessary condition for a system to be serializable. But it is unfortunately not a sufficient condition. Consider again the Lipsky/Papadimitriou example (Figure 9). The forbidden region is connected but not simply connected (it is homeomorphic to a solid torus). There is a dipath

[4]When we restrict to a path-connected component.

[5]Or in fact as we will see later on when looking at higher-order homotopies, we need a base dipath.

going through the center of this "torus" which cannot be deformed on one of the boudaries of the outer cube of states.

We definitely need some new theory here, developed in Section 6 in particular. But let us divert a little and look at another "geometrically flavoured" model for concurrency.

3. A Combinatorial Approach

Let us get back to transition systems now. In fact, there is another "geometric" model for concurrency which seems to relate more to transition systems than progress graphs, introduced in the article by Vaughan Pratt [**54**], and which has inspired the following work on the subject a lot (for instance [**26**]). It was essentially motivated by a defect in the duality between event structures and automata[6], two well known mathematical models for concurrency.

The models which have been introduced to fix this defect, which can be attributed to the fact that the former semantics is a "truly-concurrent" one where the latter is a simulation by interleaving, were based on one form or another of CW-complex. Such objects are gluings of of "elementary" shapes along their boundaries. The following explanation is inspired by [**26**].

Consider first transition systems. They allow to model concurrency with an interleaving semantics. They already are (1-dimensional) geometric objects. Many important semantic properties are actually geometric properties on their underlying graph of transitions. For instance, initial and final (or deadlocking) states, unreachables states, cycles, branchings and confluences, as seen in the introductory section. All these geometric notions are important for validating concurrent systems or proving them correct. For instance, branchings are of importance for the so-called "branching-time" semantic equivalences such as bisimulation equivalence [**50**], and deadlocks and unreachable states are useful for static analysis (such as model-checking for instance).

In fact, the modelisation of concurrent systems by interleaving naturally constructs cubical shapes. For example, squares like $a \mid b$:

$$\begin{tikzcd} \cdot \arrow[r, "a"] \arrow[d, "b"'] \arrow[dr, phantom, "A"] & \cdot \arrow[d, "b'"] \\ \cdot \arrow[r, "a'"'] & \cdot \end{tikzcd}$$

which represents the asynchronous execution of actions a and b (a' and b' are transitions which have respectively a and b as "labels").

The natural idea is that this interleaving is an expansive encoding of the fact that a and b are independent indeed. Using a physicist's image, we would like to represent all the ways we can mix together any number of sub-actions of a and b (all the shuffles of possibly infinitely many chunks of a and b) as shown in Figure 10.

[6]Which, later, will also motivate the introduction of Chu spaces [**55**].

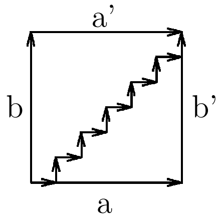


Figure 10: Possible interleavings.

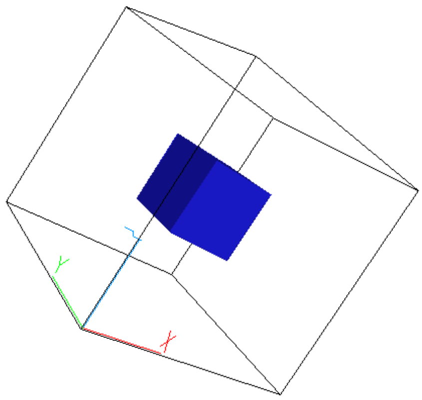


Figure 11: A precubical set representing three concurrent accesses to a 2-semaphore.

It is thus natural to put into our model all the possible subdivisions of the square, i.e., the interior A of the square as well as its boundary. Adding in the model the concept "interior of a square" (as well as "interior of any n-cube" when we model n concurrent processes in parallel) naturally leads to the notion of *precubical set*.

The usual way to define a precubical set is to define the boundary operators; for instance, for the square, we have four boundary operators, respectively corresponding to a, b, a' and b'. This is not enough since we want to encode the "direction of time" as well in this model. In dimension one, we use a "directed graph" of transitions; we would like something similar here but with "higher-dimensional" transitions.

The choice made in [**26**] is to divide the family of boundary operators into two families of operators. In the case of a square, we have a family of two source boundary operators d_0^0 and d_1^0, with $d_0^0(A) = a$ and $d_1^0(A) = b$, and a family of two target boundary operators d_0^1 and d_1^1, with $d_0^1(A) = a'$ and $d_1^1(A) = b'$. This naturally extends the notion of source and target of arcs of directed graphs.

If a 2-transition (square) is nothing but an independence relation between two 1-transitions, a 3-transition, or cube, is not merely a shortcut for 3 independence relations. The fact that action a is independent of action b, b independent of c, and c of a does not imply that a, b and c can be executed altogether in an asynchronous manner. It is the case for instance of an abstraction of a print spooler with two printers, or of two floating-point units[7], or of a communication buffer with two cells, i.e., of a 2-semaphore s.

If we use the notations of E. W. Dijkstra [**10**], it suffices to consider the three actions $a = b = c = Ps$; s can be shared by two but not by three processes at the same time (see Figure 11). This kind of object which synchronizes very weakly is of great importance for fault-tolerant distributed systems[8].

These properties cannot be expressed simply (if at all) in most of the other mathematical models for concurrency, as asynchronous transition systems, prime event structures etc. a notable exception being Petri nets. These have other drawbacks: they are not very compositional, which makes them clumsy for analyzing concurrent programs[9].

Of course this can be easily (and fruitfully) generalized. The concurrent access of $n+1$ processes to a given n-semaphore is represented by the boundary of an hypercube of dimension $n+1$. This means we need to add n-transitions to our model for all $n > 0$.

There again, we divide the family of boundary operators into two subfamilies: the set of n-transitions will have a family of n source boundary operators d_i^0, $0 \leqslant i \leqslant n-1$ (all giving $(n-1)$-transitions), and a family of n target boundary operators d_j^1, $0 \leqslant j \leqslant n-1$. For instance, for $n = 3$:

$$
\begin{tikzcd}
(0,0,0) \arrow[rr, "b"] \arrow[dr, "a"'] \arrow[dd, "c"'] & & (1,0,0) \arrow[dr] \arrow[dd] & \\
& (0,1,0) \arrow[rr] \arrow[dd] & & (1,1,0) \arrow[dd] \\
(0,0,1) \arrow[rr] \arrow[dr] & & (1,0,1) \arrow[dr] & \\
& (0,1,1) \arrow[rr] & & (1,1,1)
\end{tikzcd}
$$

The interior D of the cube has three source boundaries, the three faces containing $(0,0,0)$, and three target boundaries, the three faces containing $(1,1,1)$. Let A, B and C be the faces

$$((0,0,0),(1,0,0),(0,0,1),(1,0,1))$$

$$((0,0,0),(0,1,0),(0,0,1),(0,1,1))$$

[7]For instance in Intel microprocessors.

[8]A shared FIFO, i.e., First In First Out stack with two entries allows for instance to implement wait-free binary consensus for two processes, whereas a simple shared variable (with atomic reads and writes) does not allow it. A good reference for these problems is [**42**].

[9]They are nevertheless very much used for analyzing boolean (telecommunication) protocols for instance.

$$((0,0,0),(1,0,0),(0,1,0),(1,1,0))$$

respectively. Let A', B' and C' be the parallel faces of A, B and C respectively.

Let $d_0^0(D) = A$, $d_1^0(D) = B$, $d_2^0(D) = C$ and $d_0^1(D) = A'$, $d_1^1(D) = B'$, $d_2^1(D) = C'$. Then $d_0^0(A) = b$, $d_1^0(A) = c$, $d_0^0(B) = a$, $d_1^0(B) = c$, $d_0^0(C) = a$, $d_1^0(C) = b$.

More generally, we can prove what we see here, that is, the boundary operators can be defined so that they satisfy the following commutation rules (for $i < j$ and $k, l = 0, 1$):

$$d_i^k \circ d_j^l \quad = \quad d_{j-1}^l \circ d_i^k$$

For instance, for a 2-transition, the relation with $k = 0$, $l = 1$ and $i = 0$, $j = 1$ means that the source of the target number one (i.e., of b') is the same as the target of the source number zero (i.e., of a). This gives us the (classical, but presented in a slightly different manner) notion of precubical set:

Definition 5. *A precubical set is a graded set $M = (M_i)_{i\in\mathbb{N}}$ with two families of operators:*

$$M_n \underset{d_j^1}{\overset{d_i^0}{\rightrightarrows}} M_{n-1}$$

($i, j = 0, \ldots, n-1$) satisfying the relations

$$d_i^k \circ d_j^l \quad = \quad d_{j-1}^l \circ d_i^k$$

($i < j$, $k, l = 0, 1$)

Of course, this is linked to progress graphs (just discretize the picture of Figure 4 using squares in the trivial way), but there is more to it than one could suspect. This is partly developed in Section 5.

I used this formalization in my first article on the subject [**30**]. In fact, cubical (that we will see a bit later) and precubical sets have a quite old history. They have been used in the first developments of algebraic topology by D. Kan and later by J.-P. Serre in his thesis [**58**]. Nowadays, combinatorial algebraic topology uses simplicial sets [**45, 18**], union of simplices of all dimensions, glued along their faces. In J.-P. Serre's thesis, cubical sets were preferred to simplicial sets because, for studying fibrations, which are locally canonical projections from a cartesian product of two topological spaces to the first one, it is simpler to consider cubical sets which have good properties with respect to projections and products[10].

We can also define a (combinatorial) notion of dipath and of dihomotopy. As we will see in Section 5, they are closely linked with the (topological) notions of dipath and dihomotopy we have seen in Section 2.

Let N be a precubical set. A dipath in N is a sequence $p = (p_1, \cdots, p_k)$ of elements of N_1 such that for all i, $1 \leqslant i < k$, $d_1^1(p_i) = d_1^0(p_{i+1})$. We say that,

- $d_1^0(p_1)$ is the initial point of p,
- $d_1^1(p_k)$ is the final point of p.

---

[10] Even if a simplicial construction was published later, see for instance [**47**].

Two combinatorial dipaths are dihomotopic if we can go from one to the other by a finite number of "transpositions" of two consecutive actions. It is in fact exactly the same notion as the "partial commutative monoids" used in Mazurkiewicz trace theory [**46**] (another model for concurrency).

Consider the "contiguity" relation (or "combinatorial dihomotopy") as follows. Let $p = (p_1, \cdots, p_k)$ and $q = (q_1, \cdots, q_l)$ be two non-empty dipaths with same initial and final states. We say that p and q are elementary contiguous if $k = l$ and if there exists u, $1 \leqslant u < k$ such that,

- for all $i < u$ and $i > u + 1$, $p_i = q_i$,
- there exists $A \in M_2$ such that $d_0^0(A) = p_u$, $d_1^1(A) = p_{u+1}$, $d_1^0(A) = q_u$ and $d_0^1(A) = q_{u+1}$.

The contiguity relation is the equivalence relation (i.e., the reflexive, symmetric and transitive closure of) generated by the elementary contiguity. We will see in Section 5 that this is very close to dihomotopy in the topological space given by the geometric realization of M indeed.

To actually give semantics to concurrent systems, the use of cubical or precubical sets is natural as I already explained (for instance by starting with the "interleaving semantics" of transition systems); this is fully examplified in [**14**] for our little P, V language. The link can be made more formal as hinted at in next section.

4. Interpretation in terms of concurrency theory

Reciprocally, all "geometric shapes" built by glueing together hypercubes of any dimension *along their corresponding upper and lower faces* can be presented as a *precubical set*[11]. For this to be clear, we need a number of definitions and lemmas.

Let K and L be two precubical sets. Then $f = (f_n)_{n \in \mathbb{N}}$ is a morphism of precubical sets from K to L if for all $n \in \mathbb{N}$, f_n is a function from K_n to L_n such that:

$$f_n \circ \partial_i^\alpha = \partial^{\alpha_i} \circ f_{n+1}$$

(for all i, $0 \leqslant i \leqslant n$)

This forms a category called Υ^S. It is a presheaf category as follows. Let $\square^1$ be the free category whose objects are $[n]$, where $n \in \mathbb{N}$, and whose morphisms are generated by,

$$[n-1] \underset{\delta_j^1}{\overset{\delta_i^0}{\rightrightarrows}} [n]$$

for all $n \in \mathbb{N} \backslash \{0\}$ and $0 \leqslant i, j \leqslant n-1$, such that $\quad \delta_j^k \delta_i^l \quad = \quad \delta_i^l \delta_{j-1}^k \qquad (i < j)$.

Now, the category $\square^{1^{op}}\mathbf{Set}$ of contravariant functors from $\square^1$ to $\mathbf{Set}$ (morphisms are natural transformations) is isomorphic to the category of precubical sets. This

[11] This terminology was recently suggested to us by Ronnie Brown. Before this, we used the term "precubical set" by analogy with the old term "presimplicial set" or simplicial (formerly called complete presimplicial) sets without degeneracy operators.

implies, by general theorems [**44**], that Υ^S is an elementary topos. Moreover it is complete and co-complete because **Set** is complete and co-complete.

The category of precubical sets of dimension less or equal than n can be seen as the presheaf category $(\square^{1\leqslant n})^{op}$**Set** where $\square^{1\leqslant n}$ is the full subcategory of $\square^1$ where objects are $[p]$ with $p \leqslant n$.

Lemma 1. *Let T_n (respectively T_n^S) be the function from Υ^S to Υ_n^S, which to every $M \in \Upsilon^S$ associates $N \in \Upsilon_n^S$ with,*

$$N([k]) = M([k]) \text{ if } k \leqslant n$$

$$N(e_i : [k+1] \to [k]) = M(e_i) \text{ for } k < n$$

$$N(\partial_i^\alpha : [k-1] \to [k]) = M(\partial_i^\alpha) \text{ for } k < n$$

defines a functor, called the n-truncation functor.

Now, let $D_{[n]}$ be the representable functor from $\square^1$ to **Set** with $D_{[n]}([p]) = \square^1([p],[n])$. We define the singular n-cubes of a precubical set M to be any morphism $\sigma : D_{[n]} \to M$.

Lemma 2. *The set of singular n-cubes of a precubical M is in one-to-one correspondence with M_n. The unique singular n-cube corresponding to a n-cube $x \in M_n$ is denoted by $\sigma_x : D_{[n]} \to M$. It is the unique singular n-cube σ such that $\sigma(Id_{[n]}) = x$.*

Proof. The proof goes by Yoneda's lemma [**43**]. □

There is only one morphism in $\square^1$ from a given $[n]$ to itself, the identity of $[n]$, hence $D_{[n]}\backslash\{Id\}$ is a functor which has only as non-empty values the $D_{[n]}([p])$ with $p < n$ ("it is of dimension $n-1$"). We write $\partial D_{[n]}$ for this functor. For σ a natural transformation from $D_{[n]}$ to M, we write $\partial\sigma$ for its restriction to $\partial D_{[n]}$.

Proposition 1. *Let M be a precubical set. The following diagram is co-cartesian (for $n \in \mathbb{N}$),*

$$\begin{tikzcd}
\coprod_{x\in M_{n+1}} \partial D_{[n+1]} \arrow[r, "\bigsqcup_{x\in M_{n+1}} \partial\sigma_x"] \arrow[d, "\subseteq"'] & T_n(M) \arrow[d, "\subseteq"'] \\
\coprod_{x\in M_{n+1}} D_{[n+1]} \arrow[r, "\bigsqcup_{x\in M_{n+1}} \sigma_x"] & T_{n+1}(M)
\end{tikzcd}$$

where $\partial D_{[n+1]} = T_n(D_{[n+1]})$ and $\partial\sigma_x = \sigma_{x|\partial D_{[n+1]}}$.

Proof. We mimick the proof of [**18**]: it suffices to prove that the diagram below (in the category of sets) is cocartesian for all $p \leqslant n+1$,

$$\begin{tikzcd}
\coprod_{x\in M_{n+1}} (\partial D_{[n+1]})_p \arrow[r, "\bigsqcup_{x\in M_{n+1}}(\partial\sigma_x)_p"] \arrow[d, "\subseteq"'] & (T_n(M))_p \arrow[d, "\subseteq"'] \\
\coprod_{x\in M_{n+1}} (D_{[n+1]})_p \arrow[r, "\bigsqcup_{x\in M_{n+1}}(\sigma_x)_p"] & (T_{n+1}(M))_p
\end{tikzcd}$$

since colimits (hence pushouts) are taken point-wise in a functor category into **Set**.

For all $p < n+1$, the inclusions are in fact bijections, and the diagram is then obviously cocartesian.

For $p = n+1$, the complement of $\bigsqcup_{x\in M_{n+1}}(\partial D_{[n+1]})_p$ in $\bigsqcup_{x\in M_{n+1}}(D_{[n+1]})_p$ is the set of copies of cubes $Id_{[n+1]}$, one for each cube of M_{n+1}. This means that the map $\bigsqcup_{x\in M_{n+1}}(\sigma_x)_p$ induces a bijection from the complement of $\bigsqcup_{x\in M_{n+1}}(\partial D_{[n+1]})_p$ onto the complement of $(T_n(M))_p$. This implies that the diagram is cocartesian for $p = n+1$ as well. ☐

This lemma states that the truncation of dimension $n+1$ of a precubical set M is obtained from the truncation of dimension n of M by attaching some standard $(n+1)$-cubes $D_{[n+1]}$ along the boundary $\partial D_{[n+1]}$ of $(n+1)$-dimensional holes. In fact, any precubical set M is the direct limit of the diagram consisting of all inclusions $T_{n-1}(M) \hookrightarrow T_n(M)$, hence is also the direct limit of the diagram consisting of all the cocartesian squares above. Computer-scientifically, this means that any precubical set can be seen as a (unlabelled) transition system, which is its 1-skeleton, on which we add independence relations. Filled-in squares specify that two actions commute, i.e., that they can be run asynchronously. This is exactly the asynchronous automata sort of models [**5**], [**59**] and [**46**]. Then in higher-dimensions, we fill in cubes etc. meaning that we add some extra (n-ary) independence relations. This is fully worked out in [**29**] in the form of adjunctions between suitable categories of transition systems and of asynchronous automata with (pre-)cubical sets. In fact, to do this properly, we have two steps to make. First, it is easy to relate $\square^{1^{op}}$**Set** with unlabelled transition systems only if we take as morphisms for transition systems, the "total" morphisms of [**68**], i.e., graph morphisms; this is unfortunately not quite enough, and to add the right morphisms is reflected on the geometric side by added degeneracy operators, i.e., by going from precubical to cubical sets. Secondly, we have to add up labels to cubical sets. This can easily be done using a comma category constructions: labelled cubical sets are just labelling morphims from a "shape" cubical set to a "labelling" cubical set. In fact, it is a particular case of a sconing construction [**17**], and as a side result we automatically know that labelled cubical sets will also form a topos.

5. A Useful Generalization

Progress graphs are a very limited model for concurrency: in particular, we are unable to give a semantics to recursive processes other than unfold all loops, whereas we could give a semantics to loops in the combinatorial model. This is not very satisfying and motivates a more local definition: a first natural idea is to impose only a local partial order instead of a global one, on a topological space, leading to a definition very much like differentiable manifolds. We recap here the main definitions, the full details can be found in [**15**].

Definition 6. *Let X be a topological space. A collection $\mathcal{U}(X)$ of pairs $(U, \leqslant_U)$ of opens of X, covering X, and partially ordered by $\leqslant_U$ is called a local partial order on X if for all $x \in X$ there exists a non-empty open neighbourhood $W(x) \subset X$ such that the restrictions of $\leqslant_U$ to $W(x)$ coincide for all $U \in \mathcal{U}(X)$, i.e.,*

for all $U_1, U_2 \in \mathcal{U}(X)$ such that $x \in U_i$ and for all $y, z \in W(x) \cap U_1 \cap U_2$:

$$y \leqslant_{U_1} z \Leftrightarrow y \leqslant_{U_2} z.$$

We call the collection of opens U of the definition, an atlas for X (by analogy with differentiable manifolds). Again by analogy with differentiable manifolds, there is a notion of equivalence of atlases:

Definition 7.

- *Two local partial orders on X are "equivalent" if their union is a local partial order on X.*
- *A locally partially ordered space consists of a topological space X and of an equivalence class of local partial orders on X. If moreover there exists a covering $\mathcal{U}$ in this equivalence class such that all $(U, \leqslant_U) \in \mathcal{U}$ are po-spaces, then we say that X is a local po-space.*

Let us give a simple example. A "directed" loop $S^1 = \{e^{i\theta} \in \mathbb{C}\}$ is a local po-space: it suffices to take $U_1 = \{e^{i\theta} \in S^1 | 0 < \theta < 2\pi\}$ with the order induced by the natural order on the θ and $U_2 = \{e^{i\theta} \in S^1 | \pi < \theta < 3\pi\}$ again with the natural order on the θ.

We need now to define suitable morphisms between local po-spaces, which in turn will give the notion of dipath.

Definition 8. *Let $(X, \mathcal{U})$ and $(Y, \mathcal{V})$ be two local po-spaces.*

A continuous function $f : X \to Y$ is called a dimap *if for all $x \in X$ there exists a subset $V(f(x)) \subset Y$ on which $\leqslant_Y$ is well defined and $U(x) \subset f^{-1}(V(f(x)))$ on which $\leqslant_X$ is well defined, such that for all $y, z \in U(x)$: $\quad y \leqslant_X z \Rightarrow f(y) \leqslant_Y f(z)$.*

There again, a dipath on X is then a dimap $f : \vec{I} \to X$ where $\vec{I}$ is the topological space $I = [0, 1] \subseteq \mathbb{R}$ with the (global) partial order inherited from the one of $\mathbb{R}$. We still write $\vec{P}_1(X)$ for the set of dipaths on X and $\vec{P}_1^{\alpha,\beta}(X)$ for the set of dipaths on X going from α to β.

The notion of dihomotopy is exactly the same as for po-spaces; let f and g be two dipaths on X between an initial point α and a final point β. A dihomotopy between f and g is a dimap from $\vec{I} \times I$ to X such that for all $x \in \vec{I}$, $t \in I$, $H(x, 0) = f(x)$, $H(x, 1) = g(x)$ and $H(0, t) = \alpha$, $H(1, t) = \beta$. We write once more $f \sim g$.

If we want to be more general, we should consider maximal dipaths (with respect to an obvious "extension" partial order) and not dipaths from an initial point to a final point. This is partially developed in [**15**] but there are still a number of open problems, in particular about infinite dipaths (on local po-spaces which are not compact).

Now, we can link the (new) topological model with the combinatorial one (the sequel is also taken from [**15**]).

Let $\square_n$ be the "standard" n-cube in $\mathbb{R}^n$ ($n \geqslant 0$),

$$\square_n = \{(t_1, \dots, t_n) | \forall i, 0 \leqslant t_i \leqslant 1\}$$

$$\square_0 = \{0\}$$

and let $\delta_i^k : \square_{n-1} \to \square_n$, $1 \leqslant i \leqslant n$, $k = 1, 2$, be the continuous functions ($n \geqslant 1$),

$$\begin{tikzcd} \square_n & \square_{n-1} \arrow[l, "\delta_i^0"'] \\ \square_{n-1} \arrow[u, "\delta_i^1"] & \end{tikzcd}$$

defined by,

$$\delta_i^k(t_1, \dots, t_{n-1}) = (t_1, \dots, t_{i-1}, k, t_i, \dots, t_{n-1})$$

Given a precubical set M, consider now the set $\mathbf{R}(M) = \coprod_n M_n \times \square_n$. The sets M_n can be considered as topological spaces with the discrete topology and $\square_n$ inherits the topological structure of $\mathbb{R}^n$. Thus $\mathbf{R}(M)$ is a topological space with the disjoint union topology. Let now $\equiv$ be the equivalence relation induced by the identities:

$$\forall k, i, n, \forall x \in M_{n+1}, \forall t \in \square_n, n \geqslant 0,$$

$$(\partial_i^k(x), t) \equiv (x, \delta_i^k(t))$$

Let $\mid M \mid = \mathbf{R}(M)/\equiv$ with the quotient topology. The topological space $\mid M \mid$ is called the *geometric realization* of M.

All this is rather classical as a direct imitation of the geometric realisation functor from simplicial sets to topological spaces. The only trouble here is to find how to interpret the direction of time as it is prescribed in precubical sets (as seen in the definition of dipaths for instance) in terms of local partial orders. For this, we restrict ourselves to non-pathological situations[12].

Definition 9. *[**15**] Let M be a precubical set. We say that M is not* self-linked *if for all its n-cubes x, $\partial_l^k(x) = \partial_{l'}^{k'}(x)$ implies $k = k'$ and $l = l'$.*

Let $x \in M$. The star of x in M is

$$St(x, M) = \{y \mid \partial_{l_1}^{k_1} \dots \partial_{l_u}^{k_u} y = x\}$$

(see for instance [**65**]).

Then we set, for $y \in St(x, M)$,

$$(x, t) \leqslant_x^U (y, u) \text{ if } \delta_{l_i}^{k_i} \dots \delta_{l_1}^{k_1}(t) \leqslant u \text{ in } \square_{n+i}$$

[12] Which might well be non-necessary; this is currently been worked out.

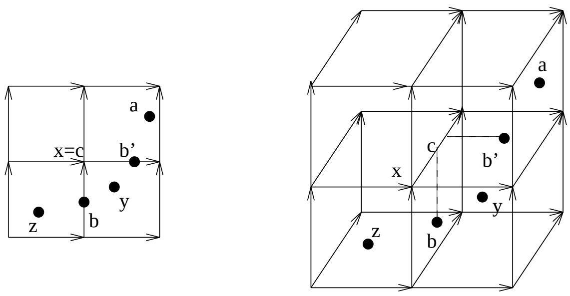


Figure 12: Illustration of the transitivity of $\leqslant_x$.

$$(y,u) \leqslant^U_x (x,t) \text{ if } \delta^{k_i}_{l_i} \dots \delta^{k_1}_{l_1}(t) \geqslant u \text{ dans } \square_{n+i}$$

Let x be an element of M and (z,v) be a point in U^x whose carrier is z. We set $(z,v) \leqslant_x (y,u)$ if there exists b in the star of x and t such that $(z,v) \leqslant^U_b (b,t) \leqslant^U_b (y,u)$.

It is a partial order indeed; the only difficulty lies in the proof of transitivity (see Figure 12). As a matter of fact, both on the left and right hand sides of the figure, we have $z \leqslant_b y$ and $y \leqslant_{b'} a$ but on the left hand side, $z \leqslant_x a$, and on the right hand side $z \leqslant_c a$ where b is the segment going from the front faces to the back faces from x.

Then, the geometric realisation of a non self-linked precubical set M defines a local po-space with atlas $\{\mathrm{St}(x,M)/x \in M_0\}$ and local partial orders $\leqslant_x$ on each $\mathrm{St}(x,\, M)$.

The geometric realisation is functorial, I refer the reader to [**15**]. We also have a right-adjoint to this functor, which is a "singular cube functor" defined very similarly as in simplicial sets theory.

The correspondence between homotopical properties in the topological case and in the combinatorial (cubical) case fortunately looks quite nice. This implies that we will be able to reason about concurrent systems both geometrically on local po-spaces (for instance on progress graphs) and algebraically or combinatorially on cubical sets.

This has at least been proven in the simpler case of dimension two in [**15**]. The geometric realisation of a combinatorial dipath p of M induces a (topological) dipath $\mid p \mid$ in $\mid M \mid$. Every combinatorial dihomotopy between p and q in M induces a (topological) dihomotopy between $\mid p \mid$ and $\mid q \mid$.

We also have the inverse one could hope for. Let L be a finite precubical set and h be a dipath in $\mid L \mid$ (i.e., a dipath from $\square_1$ to $\mid L \mid$). Then there exists a "cubical approximation" $f : S_k \to L$ of h where S_k is a subdivision of $\vec{I}$. Moreover f defines a (combinatorial) dipath $(f(u_1), \dots, f(u_k))$ which we call $\tilde{f}$. Finally, $\mid f \mid$ is homotopic to $\mid \tilde{f} \mid$ in $\mid L \mid$.

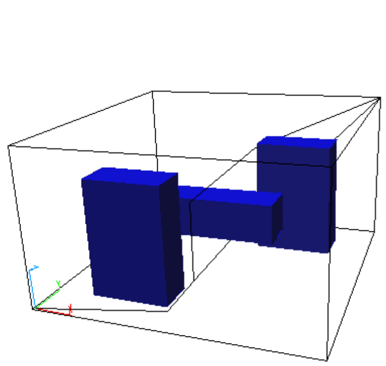


Figure 13: "A room with 3 barriers" and two non dihomotopic dipaths.

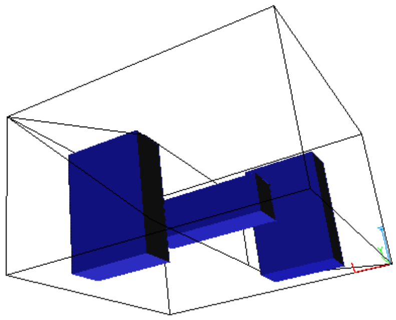


Figure 14: "A room with 3 barriers" (another view).

6. First study of dihomotopy

We have already seen that the equivalence classes of dipaths modulo dihomotopy are less numerous in general than the homotopy classes of dipaths. Since the article [**54**], as well as in [**26**], I had the intuition that studying the dihomotopy classes of dipaths with fixed extremities was equivalent to studying the homotopy classes of dipaths with fixed extremities. In fact this is not true. It suffices to consider the example of Figures 13 and 14 which give semantics to terms[13]

```
#sem c 2
A=Pa.Pc.Va.Pb.Vc.Vb
B=Pa.Va.Pc.Vc.Pb.Vb
C=Pc.Vc
PROG=A|B|C
```

The two dipaths that are represented on these pictures are homotopic but not dihomotopic. The two dipaths do correspond to real executions of a simple concurrent program (c being a very simple 2-place buffer, a and b being two shared scalar variables). This implies we need new tools.

In fact, it is easy to build an analogue of a fundamental groupoïd, which will only be a fundamental category in fact. It is obviously linked to the construction of diconnected sets [**15**], and also to the recent constructions of S. Sokolowski (see [**64**]), but there is still some work to be done in this direction (see [**57, 31**]).

Let $\mathcal{X} = (X, \mathcal{U}, (\leqslant_U)_{U \in \mathcal{U}})$ be a local po-space. We can define as usual, a composition operation between some of the dipaths of X, going from $\vec{P}_1^{\alpha,\beta}(X) \times \vec{P}_1^{\beta,\gamma}(X)$

[13] `#sem 2` means that `c` is a 2-semaphore. I have used in the sequel the syntax that my toy static analyzer uses (see `http://www.di.ens.fr/~goubault/analyse.html`).

to $\vec{P}_1^{\alpha,\gamma}(X)$ (with the same notations as we had with po-spaces, i.e., $\vec{P}_1^{\alpha,\beta}(X)$ is the set of dipaths from α to β). It is not a commutative nor associative operation in general, but neither is it in the classical case. Similarly to the classical case, this operation induces a composition operation of classes of dipaths modulo dihomotopy and then becomes associative.

We can then define the following category $\vec{\pi}_1(X)$:

- its objects are the points of X,
- its morphisms are the dihomotopy classes of dipaths: a morphism from x to y is a dihomotopy class $[f]$ of a dipath f going from x to y.

The composition defined earlier is an associative operation with identities (the dihomotopy classes of constant dipaths), we use it as the composition of morphisms.

In fact, this construction even defines a functor from the category of local po-spaces to the category of categories. Let $f : X \to Y$ be a dimap from a local po-space X to a local po-space Y. We define $\vec{\pi}_1(f)$ as a morphism from $\vec{\pi}_1(X)$ to $\vec{\pi}_1(Y)$:

- on objects x of $\vec{\pi}_1$, $\vec{\pi}_1(f)(x) = f(x)$,
- on morphisms $[\omega]$ of $\vec{\pi}_1$, with ω any dipath, $\vec{\pi}_1(f)([\omega]) = [f \circ \omega]$.

We have introduced in [**15**] the notion of “ diconnected components” to study the dihomotopy classes of dipaths. This should be the natural counterpart of connected components in usual topology, but as the relation xRy if there exists a dipath from x to y is certainly not an equivalence relation (and we certainly do not want to make it symmetric since this would mean we would study the arcwise connected components), this is more complicated. In fact, this is linked to a certain category of fractions of the fundamental category, see [**31**].

Definition 10. *1. The* homotopy history *of a maximal dipath*[14] $\alpha : I \to X$ *is*

$$h\alpha := \{ \quad y \in X | \exists \text{ a dipath } \beta \text{ going through } y \text{ and } \alpha \sim \beta \}$$

2. Two points are homotopy history equivalent *of*

$$x \in h\alpha \Leftrightarrow y \in h\alpha \text{ for all } \alpha \in \vec{P}_1(X).$$

3. The diconnected components *of X are the connected components (in the classical sense) of the equivalence classes of dipaths modulo the homotopy history equivalence in X.*

For instance, the complement of the “Swiss flag” in $\vec{I}^2$ (see Figure 15) has 10 diconnected components. This example gives semantics to the program having as parallel processes $T_1 = Pa.Pb.Vb.Va$ and $T_2 = Pb.Pa.Va.Vb$ (where a and b are 1-semaphores).

In region 1, we have all possible futures. In region 2, we can only go in the future to regions 4 and 6, i.e., the program will deadlock or T_2 will access to a and b before T_1. In region 6, we can only have come from 2 and go to 9: T_2 accesses a and b

[14] When this exists, for instance when the space is compact.

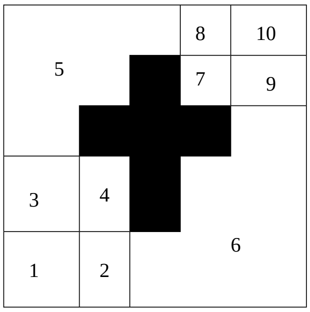


Figure 15: The "Swiss flag".

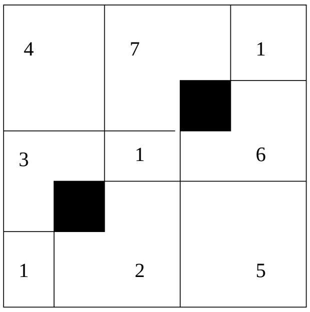


Figure 16: "Two ordered holes".

before T_1. In region 9, we can have "come" from the unreachable region 7 or from 6. In region 10, we can have come from any other region except 4.

The complement of the "two ordered holes" in $\vec{I}^2$ (see Figure 16), which gives semantics to $Pa.Va.Pb.Vb \mid Pa.Va.Pb.Vb$) has 7 classes modulo the homotopy history equivalence. One of these contains both the initial point **0**, the final point **1**, and a region in the center of the Figure. This class is decomposed into 3 diconnected components.

The "room with 3 barriers" example in $\vec{I}^3$ (see Ex. 13) has 8 classes modulo the homotopy history equivalence. One of the classes (in the center) is decomposed into two diconnected components.

This point of view has in particular made it possible to prove that the "2-phase locking" protocol, which regulates the access to fields of a distributed database is correct (i.e., is sequentialisable). We can find this proof, by M. Raussen, based on ideas of [**35**], in the article [**15**]. The reader should notice that S. Sokolowski has defined in [**64**] a quite similar point of view (about diconnected components) but without discriminating the future of dipaths. This can be more interesting in some situations (one of which might be when studying bisimulation equivalence, [**50**]). The interested reader can look at his other papers, [**62**], [**63**] and [**61**].

The problem now is to get to calculate or characterize somehow these dangerous regions etc. In Section 7 I discuss some ideas that have been used for this purpose. It is to be noted that in the case of the regular expressions we had at the beginning, we have a "critical point" approach to obstructions to dihomotopy, which is very algorithmic in nature. We refer the reader to [**14**], [**13**] and [**16**] (for the detection of deadlocks and unsafe regions), and [**56**] (for the classification of dipaths modulo dihomotopy in such models). Let us concentrate on the more general models in the following section.

7. Dihomotopy invariants

7.1. Homology

In algebraic topology, it is well known that homotopy is a subtle notion. It is in general very hard to prove that two topological spaces are homotopy equivalent, i.e., that one is an "elastic" deformation of the other. Even homotopy groups, whose isomorphism is necessary and sometimes sufficient[15] to decide of the homotopy equivalence are extremely hard to compute.

Nevertheless, there exist so called "homotopy invariants" which can be computed. A homotopy invariant is a functor which to every topological space (or one in a suitable sub-category) X associates a mathematical object $S(X)$ such that if X and Y are homotopically equivalent $S(X)$ and $S(Y)$ are isomorphic.

My first idea, expressed in [**30**], was that it was better to consider some homological invariants to compute important properties of concurrent and distributed systems.

To begin with, we can make a very simple remark: instead of starting with a concurrent program semantics expressed in the form of precubical sets $M = (M_i)_{i\in\mathbb{N}}$, we can use "bi-graded" precubical sets, i.e., sets

$$N = (N_{p,q})_{p,q\in\mathbb{N}}$$

The start boundary operators d_i^0 are now going from $N_{p,q}$ to $N_{p-1,q}$ and the end boundary operators d_i^1 are going from $N_{p,q}$ to $N_{p,q-1}$. The sets $N_{p,q}$ are disjoint only if the shape it models does not contain any "directed" cycle.

The crucial observation is:

Lemma 3. *[**26**] Consider the following diagram of R-modules (R being an integral domain, for instance $\mathbb{Z}$ or $\mathbb{Z}/2\mathbb{Z}$ as in [**30**]):*

$$\mathcal{A}(N) = \quad \begin{array}{ccc} \mathcal{A}(N_{p,q}) & \xrightarrow{\partial_0} & \mathcal{A}(N_{p-1,q}) \ldots \\ \partial_1 \big\downarrow & & \vdots \\ \mathcal{A}(N_{p,q-1}) \ \ldots & & \end{array}$$

where $\mathcal{A}(N_{p,q})$ is the free R-module generated by $N_{p,q}$ and[16]

$$\partial_0 = \sum_{i=0}^{i=p+q-1} (-1)^i d_i^0$$

$$\partial_1 = \sum_{i=0}^{i=p+q-1} (-1)^i d_i^1$$

It is a ("weak") bicomplex of modules, i.e., it satisfies the equalities: $\partial_0 \circ \partial_0 = 0$, $\partial_1 \circ \partial_1 = 0$, et $\partial_0 \circ \partial_1 + \partial_1 \circ \partial_0 = 0$.

For instance, the automaton:

---

[15] When the isomorphism is induced by a continuous map between CW-complexes for instance.

[16] In order to remember the dimension in which they act, we will sometimes write ∂_0^{p+q} and ∂_1^{p+q}.

$$\begin{tikzcd}
& \beta \arrow[dr, "a'"] & \\
1 \arrow[ur, "b"] \arrow[dr, "a"'] & & \gamma \\
& \alpha \arrow[ur, "b'"'] &
\end{tikzcd}$$

is represented by the bicomplex of $\mathbb{Z}$-modules,

$$\begin{tikzcd}
& (a)\oplus(b) \arrow[r, "\partial_0"] \arrow[d, "\partial_1"'] & (1) \arrow[d, "\partial_1"'] \\
(a')\oplus(b') \arrow[r, "\partial_0"] \arrow[d, "\partial_1"'] & (\alpha)\oplus(\beta) \arrow[r, "\partial_0"] \arrow[d, "\partial_1"'] & 0 \arrow[d, "\partial_1"'] \\
(\gamma) \arrow[r, "\partial_0"] & 0 \arrow[r, "\partial_0"] & 0
\end{tikzcd}$$

with $\partial_0(a) = \partial_0(b) = 1$, $\partial_1(a) = \partial_0(b') = \alpha$, $\partial_1(b) = \partial_0(a') = \beta$ and $\partial_1(a') = \partial_1(b') = \gamma$.

The bicomplexes (or double complexes of modules) are very important objects in homology, they have in fact very many interesting properties.

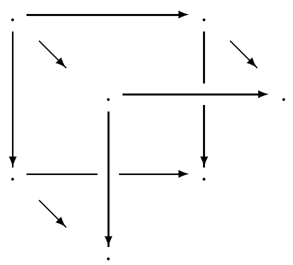
Let,

- $H_i(N, \partial_0) = \frac{Ker\ \partial_0^i}{Im\ \partial_0^{i+1}}$
- $H_i(N, \partial_1) = \frac{Ker\ \partial_1^i}{Im\ \partial_1^{i+1}}$

where *Ker* f (respectively *Im* f) is the kernel (respectively the image) of the linear application f. These form the "horizontal" (respectively "vertical") homology groups, which enable to determine the branchings (respectively confluences) of the automata.

In the case of our example, we find easily, $H_0(M, \partial_0) = (\alpha)$, $H_0(M, \partial_1) = (1)$, $H_1(M, \partial_0) = (b - a)$, $H_1(M, \partial_1) = (b' - a')$, and the other homology groups are trivial. The generator $(b - a)$ of the horizontal homology group of dimension one expresses the fact that there is a non-deterministic choice between actions a and b. The generator of the first vertical homology group $(b' - a')$ shows that there is a confluence between actions a' and b'.

A typical branching in dimension two is for instance:



where the three faces are filled in.

The homology functors have nice computational properties (colimits, tensor product [Künneth formula], Mayer-Vietoris long exact sequence etc.) which enables to compute them inductively on the syntax of a parallel program, as was done in [**26**] with the process algebra CCS.

The "total homology" functor, defined as the homology functor for the boundary operator $\partial_0 - \partial_1$ gives also a homology theory for dipaths with fixed extremities modulo dihomotopy (see [**26**] and [**27**]) - in fact unfortunately, it is also an invariant of dipaths with fixed extremities modulo homotopy. This implies for instance that these functors cannot separate the two dipaths of Example 13.

To correct this defect, it is necessary, first of all, to start with a better category of cubical sets. Most of this part has been based on earlier work by R. Brown and P. Higgins [**7**] and [**6**], and has been developed later by P. Gaucher. I will briefly come back to this work in Section 8.2.

The theory I proposed in [**27**] defines homology groups in all dimensions indeed. The goal was to be able to distinguish between the shape of Figure 11 representing a 2-semaphore which is accessed by three processes, with a 3-semaphore in the same situation. The difference between a 1-semaphore and a 2-semaphore which two processes try to access can be noticed by examining the dihomotopy classes of dipaths. In the first case, there is necessarily a mutual exclusion which serializes the access to the shared object. In the second case, there is no serialization. When we use n-semaphores with $n > 1$, we cannot distinguish the difference of behaviour by looking at whether two consecutive actions (locks or unlocks) commute or not. The only way is by looking at the difference of behaviour when there are at least $n + 1$ consecutive actions (accesses to the semaphore) in general.

In order to spot this difference geometrically, we must examine hypersurfaces of dimension n modulo dihomotopy instead of just dipaths modulo dihomotopy. There again, we must be cautious to fix the extremities of these hypersurfaces.

The idea of [**27**] was simple and can be found in different forms in more recent work by S. Sokolowski [**64**] and P. Gaucher [**21**]. As can be seen on Figure 17 (at the left hand side, in dimension 2, at the right hand side in dimension 3) by taking two dihomotopic dipaths having the same source and target, we can consider the surfaces on which we can deform one of these paths into the other by a homotopy. We then say that two such surfaces are dihomotopic if there exists a dihomotopy between each of the dihomotopies that define these surfaces. In Figure 17 the two surfaces (one above the hole, the other below) are not continuously deformable one into the other, whereas they would be if the cube were entirely filled in. Homologically (as in [**27**]), we can look at the surfaces with fixed boundaries modulo the total homology. We briefly get back to this, homotopically, in Section 9.

7.2. Achronal cuts

In the previous section, we have tried to go from a classification problem of dipaths modulo dihomotopy to a simpler classification problem of dipaths modulo homology.

There is of course another natural idea [**15**], that of taking "instant snapshots" of the dynamics of dipaths and observe their evolution in time. In fact, this is fairly close to methods used in fault-tolerant distributed systems theory [**36**].

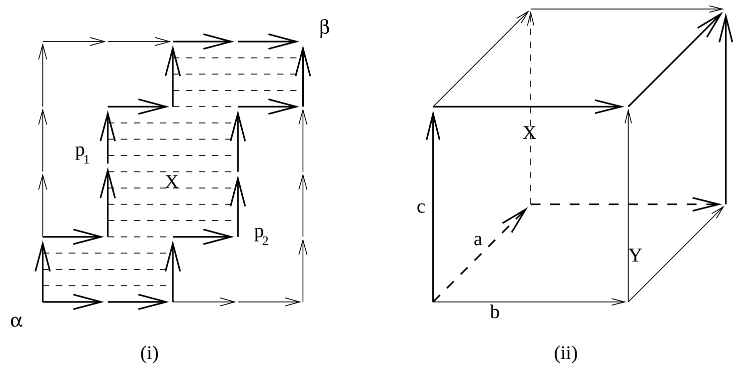


Figure 17: Two surfaces having the same boundaries which are not dihomotopic.

Definition 11. *Let $(U, \leqslant)$ be a partially ordered set. A subset $V \subset U$ is called* achronal *if for all $x, y \in V : x \leqslant y \Rightarrow x = y$ (similarly to the notion in [**53**]).*

Definition 12. *[**15**] Let $(X, \leqslant)$ be a po-space.*

1. *$(X, \leqslant)$ is a* parameterized *po-space if there exists a dimap $F : X \to \mathbb{R}$ such that $X_t := F^{-1}(t)$ is achronal for all $t \in \mathbb{R}$.*
2. *$\leqslant$ is* Euclidean*, if there exists a finite number of dimaps $f_i : X \to \mathbb{R}$ such that*

$$\forall x, y \in X : \quad x < y \Leftrightarrow \forall i : \quad f_i(x) \leqslant f_i(y);$$

$$\exists i : \quad f_i(x) < f_i(y).$$

3. *A local partial order on a topological space X is* parameterized*, respectively* Euclidean *if (one of its refinements) is a parameterized po-space, respectively, Euclidean po-space.*

A Euclidean partial order is in fact a transcription of the natural componentwise ordering on an $\mathbb{R}^n$ (like the progress graphs we saw at the beginning): given two points $\mathbf{x} = [x_1, \ldots, x_n], \mathbf{y} = [y_1, \ldots, y_n] \in \mathbb{R}^n$,

$$\mathbf{x} \leqslant \mathbf{y} \Leftrightarrow \forall i : \quad x_i \leqslant y_i.$$

In a parameterized po-space, we can always reparameterize the dipaths and the dihomotopies, such that, for any dipath p and any $t \in \vec{I}$, $p(t) \in F^{-1}(t)$.

Let $H : J \times I \to X$ be a well-parameterized dihomotopy between two well-parameterized dipaths $\alpha, \alpha' : \vec{I} \to X$. Then for all $t \in \vec{I}$, $\alpha(t)$ and $\alpha'(t)$ are in the *same* connected component of X_t (which is the "cut at instant t of X).

This gives us a means, by the study (with standard homotopy theory) of cuts, to determine the possible obstructions to dihomotopy, thus to find a subset of the possible schedules.

Unfortunately, this only gives us an approximation in general; let X be the subset $[0,3]\times[0,3]\times[0,3]\setminus[1,2]\times[1,2]\times[0,3]$ of $\mathbb{R}^3$ with the natural partial order. There are two dihomotopy classes of dipaths going from $(0,0,0)$ to $(3,3,3)$, but the cuts induced by the "height function" $F(x,y,z)=x+y+z$ are all connected.

Thus, to find all information about dihomotopy classes of dipaths, it is not enough to consider only one family of cuts. In fact, it seems that we need all possible families of cuts, in the general case of precubical sets. On the computer scientific side, this only means that some asynchronous systems have no global clock.

8. Refinements of the framework

8.1. The ω-categorical point of view

The idea goes back to the article [**54**], and has been improved by P. Gaucher. The interested reader can also read [**8**] where similar ideas, using pasting schemes, have been developed.

A ω-category is a category with morphisms and compositions in all dimensions, somehow coherent with one another. The idea for the modelisation of concurrency is that objects, or 0-morphisms, represent states of an HDA, the 1-morphisms represent all possible paths of executions (all dipaths), and the higher-dimensional morphisms represent the dihomotopies between morphisms of lesser dimension[17]. In particular, the 2-morphisms represent the dihomotopies between paths of execution. The composition laws between higher-dimensional morphisms characterize the compositions of dihomotopies of higher dimension.

As V. Pratt already noticed in [**54**], the axioms of ω-categories encode the composition properties of dipaths and of dihomotopies in an HDA. The interested reader can find the exploitation of these ideas in [**20**], [**19**] and [**22**].

P. Gaucher in [**22**] and [**21**] uses the ω-category generated by a cubical set to construct three homological theories, corresponding respectively to branchings, confluences and globes (or, computer-scientifically, mutual exclusions). This is constructed through suitable nerve functors. This is made possible because of the choice of a nice category of cubical sets first, and also because simplicial sets can be represented as ω-categories as well (see [**67**]). In some ways, the branching and confluence nerves describe simplicially all achronal cuts of HDA, as hinted at in Section 7.2.

These constructions have a number of advantages over the ones of [**26**]:

- They are more discriminating (for instance, the "room with three barriers" example should be fully described by these homology theories)
- Concerning the branching and confluence homologies, they are not sensitive to subdivision.

We should mention two other points: ω-categories seem to give the right structure to the "dihomotopy sets" or at least what should be the algebraic counterpart in the directed theory to the homotopy groups. The equivalence of categories between the category of cubical categories with connections and compositions and the category of ω-categories (see [**12**]) is certainly a step in this way. Other papers by R. Brown

[17] Precisely those introduced in Section 7.1.

and P. Higgins (see [**7**] and [**6**]) also pave the way toward Seifert/van Kampen theorems in the directed theory, probably in weaker forms though (I develop this a little in Section 9).

8.2. Cubical sets

There exist several types of cubical sets. The first important remark is that, in the category of precubical sets, morphisms are somehow too rigid.

Mathematically, it is easily seen that morphisms respect the dimension of cells and length of dipaths (the number of segments they are composed of); this means in particular that the orthogonal projection of a filled-in (and also of a hollow) square onto one of its segments is not a morphism in this category.

Computer-scientifically, this implies that some important properties are not "natural". We have to introduce some degeneracy operators; basically, what we need here is to be able to consider any m-transition or hypercube of dimension m as a n-transition ($n \geqslant m$). In computer scientific terms, the degeneracy operators will allow us to consider any execution of m busy processors as an execution of n busy processors where $n-m$ processors are busy$\cdots$doing nothing. In dimension one, this is directly connected to the notion of "idle transition" in transition systems theory, see [**68**] and [**29**].

Definition 13. *A cubical set K is a precubical set (K, ∂_i^α) with degeneracy operators $\epsilon_i : K_{n-1} \to K_n$ $(0 \leqslant i \leqslant n-1)$ verifying the relations:*

$$\begin{array}{rcll} \epsilon_i\epsilon_j & = & \epsilon_{j+1}\epsilon_i & (i \leqslant j) \\ \partial_i^\alpha \epsilon_j & = & \left\{ \begin{array}{ll} \epsilon_{j-1}\partial_i^\alpha & (i < j) \\ \epsilon_j \partial_{i-1}^\alpha & (i > j) \\ Id & (i = j) \end{array} \right. & \end{array}$$

R. Brown and P. Higgins have added later on [**7**] other special degeneracy operators called connections.

Also, we can define gluings of n-cubes or compositions, which can also reveal important to the computer-scientific modelisation, if we want to be able to consider dipaths algebraically (which are gluings of n-cubes indeed); they have been defined as well by R. Brown and P. Higgins in [**7**].

From this, one could construct ω-cubical categories, which have been shown equivalent quite recently to (see [**12**]) the category of ω-categories, whose use for the modelisation of concurrent processes has been finally put together by P. Gaucher [**22**].

In order to link the ω-categorical formulation of P. Gaucher, we have been compelled to restrict the category of local po-spaces to consider. In fact, as in standard algebraic topology, the category of topological spaces is far too big and contains far too pathological elements to be the right object of study. In general, we restrict ourself to topological spaces which have the homotopy type of a CW-complex [**41**]. P. Gaucher and myself have introduced in [**23**] a particular kind of CW-complex, which we called globular CW-complex, and which is essentially a CW-complex whose n-cells are all directed. A n-cell is homeomorphic to $\vec{I} \times I^{n-1}$ quotiented by relations

$$(k, x_1, \ldots, x_{n-1}) = (k, y_1, \ldots, y_{n-1})$$

($k = 0, 1$). The advantages of this category of local po-spaces are:

- It contains only the classical homotopy types,
- It allows to construct the homology theories of P. Gaucher (defined originally on ω-categories) in a topological framework,
- It permits to define a notion of dihomotopy equivalence (which refine the usual homotopy equivalence, i.e., the homotopy types). We hope that for the globular CW-complexes, this should coincide with a notion of weak dihomotopy equivalence (as in the classical case).

8.3. Domain theory

There are links between the theories briefly described before and domain theory[18] (for instance as developed in chapter VII of [**38**]), or of other older topological considerations (the book [**52**] for instance).

L. Nachbin in [**52**] has studied some particular kind of topological spaces, the so-called compact order-Hausdorff topological spaces. In fact, this is nothing but compact po-spaces (for instance, finite progress graphs). One of the very interesting results in the theory is that there is an adjunction between these compact po-spaces and another type of topological space (no order there!), the stably-compact spaces.

We will write PO for the category of compact po-spaces. We need an intermediary definition before defining the stably-compact spaces:

Definition 14. *[38]*

- *A closed subset Q of a topological space (X, τ) is said to be irreducible if Q is not the union of two proper closed subsets of Q,*
- *A subset S of a topological space (X, τ) is saturated if it is the intersection of open subsets of (X, τ) containing S,*
- *(X, τ) is sober if for all irreducible subsets Q of (X, τ), there is a unique $x \in X$ such that the closure of $\{x\}$ is Q.*

We now define stably-compact spaces:

Definition 15. *[**11, 39**] A stably-compact space is a topological space (X, τ) which satisfies:*

- *(X, τ) is sober,*
- *(X, τ) is compact and locally compact,*
- *The intersection of two compact saturated subsets of (X, τ) is a compact saturated subset of (X, τ).*

In fact, see [**40**], a stably-compact space is a set X together with a topology τ on X such that there exists another topology on X, τ^* on X satisfying the following conditions:

- $\tau \cup \tau^*$ is compact,

[18] This is part of a talk delivered by the author in Dagstuhl seminar 00231/1 "Topology in Computer Science".

- for all $x \not\leqslant y$[19] in X, there exists an element $O \in \tau$, an element $O^* \in \tau^*$ such that $x \in O$, $y \in O^*$ and $O \cap O^* = \emptyset$.

In some ways, (X, τ) is compact Hausdorff *with the help of the topology* τ^*.

We write *SK* for the category whose objects are the stably-compact spaces and whose morphisms are continuous functions.

Proposition 2. *[**11, 39**] Let $(X, \tau, \leqslant)$ be a compact po-space (τ is the topology, $\leqslant$ is the partial order). We build (X', τ') a topological space from $(X, \tau, \leqslant)$ as follows:*

- *$X' = X$,*
- *τ', the set of opens, is composed of elements U of τ which are such that $\forall x \in U$, $\forall y \geqslant x$, $y \in U$ ("upper sets").*

Then (X', τ') is a stably-compact space.

Sketch of proof. It is a direct consequence of the local convexity theorem (see [**52**] or [**38**], Theorem 1.4, Chapter VII, page 272) which states that sets of the form $U \cap V$, where U is an upper open set and V is a lower open set, form a base for the topology of X. Thus it suffices to take for τ^* the set of lower open sets. The axiom of "weak separation" is exactly corollary 1.2, Chapter VII, page 271 of [**38**]. □

Of course, (X', τ') is in general not at all Hausdorff.

Dimaps between compact po-spaces are naturally mapped under this transformation onto continuous maps between their stably-compact counterparts. This defines a functor α from $\mathcal{PO}$ to $\mathcal{SK}$ which has a right-adjoint we will briefly describe:

Definition 16. *[**11, 39**] Let (X, τ) be a topological space, A the set of its compact saturated subsets, A^* the set of complements (in the powerset $\wp(X)$ of X) of elements of A. The "patch" topology on X is the topology $\kappa(\tau)$ generated by the base $C = \{U \cap V \mid U \in \tau, V \in A^*\}$.*

Proposition 3. *[**11, 39**] Let (Y, σ) be a stably-compact space. We can associate with it, a structure $(X, \tau, \leqslant) = \gamma(Y, \sigma)$ with,*

- *$X = Y$,*
- *$\tau = \kappa(\sigma)$,*
- *for all $x, y \in X$, $x \leqslant y$ if for all $U \in \sigma$ with $x \in U$, y is also in U[20].*

Then $(X, \tau, \leqslant)$ is a compact po-space.

Moreover, γ defines a functor from *SK* to *PO* (transforming continuous functions *SK* into dimaps of *PO*).

Consider now a dihomotopy H between two dipaths f and g with the same source and target in X. It is simply a dimap from $I \times \vec{I}$ to X such that $H(0, .) = f$ and $H(1, .) = g$.

[19] Where $\leqslant$ is the specialization ordering of the topology τ, i.e., $x \leqslant y$ if for all $O \in \tau$, $x \in O \rightarrow y \in O$.

[20] So Y is Hausdorff is and only if $\leqslant$ is the trivial order.

Therefore $\alpha(H)$ is a continuous function from $\alpha(f)$ to $\alpha(g)$, which are themselves continuous functions from $\alpha(I\times\vec{I})$ to $\alpha(X)$. But $\alpha(I\times\vec{I}) = I\times\alpha(\vec{I})$, because $\alpha(I) = I$ (all open set of I is upper). We conclude that $\alpha(H)$ is a (classical) homotopy between $\alpha(f)$ and $\alpha(g)$. We thus have proved a simple obstruction criterion to dihomotopy:

Proposition 4. *Let f and g be two dipaths in the compact po-space X. Then $\alpha(f)$ and $\alpha(g)$ are not homotopic (in the classical sense) implies that f and g are not dihomotopic.*

Next question is, do we have a reciprocal to this? Let H' be a homotopy between $\alpha(f)$ and $\alpha(g)$. At what condition is there a dihomotopy H between f and g? Do we have $\alpha(H) = H'$?

The first natural idea to characterize the homotopies between $\alpha(f)$ and $\alpha(g)$, given f and g two dipaths in a compact po-space X, is to study the group homomorphisms between the homotopy groups of $\alpha(\vec{I})$ to the homotopy groups of $\alpha(X)$. First of all, we notice that $\alpha(\vec{I})$ is a compact and connected topological space.

As $\alpha(\vec{I})$ is connected, we can define its fundamental group $\pi_1(\alpha(\vec{I}))$ by choosing any basepoint, for instance 0. Unfortunately, the study of continous functions from I to $\alpha(\vec{I})$, reveals that they are the lower semi-continuous maps from I to I and that $\alpha(\vec{I})$ is a simply-connected topological space. This means that there is no interesting group homomorphism to look for there from $\pi_1(\alpha(\vec{I}))$ to $\pi_1(\alpha(X))$.

Even worse, we can show that we can deform in a continuous manner (for the topology of $\alpha(\vec{I}\times\vec{I})$) any maximal continuous path (for the topology of $I\times I$) in any other one, by going through discontinuous ones for the topology of $I\times I$. This entails that the homotopy between functions from $\alpha(\vec{I})$ to $\alpha(\vec{I}\times\vec{I})$ does not even enable us to see the presence of a hole in the surface $I\times I$!

In fact, the adjunction between stably-compact spaces and compact po-spaces can be changed into an equivalence of categories. The way to do this is to consider the subcategory of stably-compact spaces where we impose that the morphisms be "perfect", see [**39**]. Perfect maps are maps which are such that the inverse image of a compact saturated set is a compact saturated set[21]. The question here is, can we develop a practical homotopy theory on stably-compact spaces, with morphisms somewhere between continuous functions and perfect maps, that would give us enough information on dihomotopy? The other important question is: is there a similar counterpart to the local po-spaces? I suspect that there are also strong links with work by M. Grandis (see for instance [**33**]).

9. Some Desired Properties

Continuing our tour of classical notions that would be useful for concurrency theory, it is natural to ask ourselves what should be the counterpart of homotopy

[21]This looks very much like proper maps, since the inverse image of a saturated set by a continuous map is always saturated. The problem is that, as we are in a non-Hausdorff situation, proper maps are not exactly the maps such that the inverse image of a compact is compact. They are, see [**4**], closed maps such that the inverse image of singletons are compact sets.

exact sequences etc. in the directed case.

9.1. Seifert/van Kampen

One of the very useful theorems in the classical theory is Seifert/van Kampen's theorem which relates the fundamental group of a space which is the (not necessarily disjoint) union of two subspaces with the fundamental groups of these two subspaces, under some mild hypotheses. This would have very interesting applications for the "modular" or "compositional" analysis of concurrent systems.

Theorem 1. *Let X be a local po-space, X_1 and X_2 two local po-spaces such that,*

- $X = \mathring{X}_1 \cup \mathring{X}_2$,
- *All continuous paths (not dipaths in general) in $\mathring{X}_1 \cap \mathring{X}_2$ are concatenations of a finite number of dipaths and anti-dipaths ("zig-zag paths").*

Let $j_1 : X_1 \cap X_2 \to X_1$ (respectively $j_2 : X_1 \cap X_2 \to X_2$) and $i_1 : X_1 \to X$ (respectively $i_2 : X_2 \to X$) be the natural inclusion morphisms. Then the following diagram,

$$\begin{tikzcd}
\vec{\pi}_1(X_1 \cap X_2) \arrow[r, "\vec{\pi}_1(j_1)"] \arrow[d, "\vec{\pi}_1(j_2)"'] & \vec{\pi}_1(X_1) \arrow[d, "\vec{\pi}_1(i_1)"] \\
\vec{\pi}_1(X_2) \arrow[r, "\vec{\pi}_1(i_2)"'] & \vec{\pi}_1(X)
\end{tikzcd}$$

is co-cartesian in the category of categories.

Proof. Consider first the following commutative diagram,

$$\begin{tikzcd}
\vec{\pi}_1(X_1 \cap X_2) \arrow[r, "\vec{\pi}_1(j_1)"] \arrow[d, "\vec{\pi}_1(j_2)"'] & \vec{\pi}_1(X_1) \arrow[d, "\vec{\pi}_1(i_1)"] \arrow[ddr, "f_1"] & \\
\vec{\pi}_1(X_2) \arrow[r, "\vec{\pi}_1(i_2)"'] \arrow[drr, "f_2"'] & \vec{\pi}_1(X) & \\
& & G
\end{tikzcd}$$

where G is a category. The question is whether one can complete this commutative diagram into the following one,

$$\begin{tikzcd}
\vec{\pi}_1(X_1 \cap X_2) \arrow[r, "\vec{\pi}_1(j_1)"] \arrow[d, "\vec{\pi}_1(j_2)"'] & \vec{\pi}_1(X_1) \arrow[d, "\vec{\pi}_1(i_1)"] \arrow[ddr, "f_1"] & \\
\vec{\pi}_1(X_2) \arrow[r, "\vec{\pi}_1(i_2)"'] \arrow[drr, "f_2"'] & \vec{\pi}_1(X) \arrow[dr, "f"] & \\
& & G
\end{tikzcd}$$

Looking at the diagram on objects of these categories, everything boils down to the existence of a push-out in the category of sets, which of course holds. On the objects, define f to be (for $x \in X$),

$$f(x) = \begin{cases} f_1(x) & \text{if } x \in X_1 \\ f_2(x) & \text{if } x \in X_2 \end{cases}$$

Now, let $u : \vec{I} \to X$ be a dimap. $u^{-1}(\overset{\circ}{X_1})$, $u^{-1}(\overset{\circ}{X_2})$ form a open covering of $\vec{I}$. Let δ be its Lebesgue number, and let $0 = t_0 < t_1 < \cdots < t_{k+1} = 1$ be a subdivision of the unit interval such that we have $\mid t_{\alpha+1} - t_\alpha \mid < \delta$ for all $\alpha \in \{0, \cdots, k\}$. By definition of Lebesgue numbers, for all $\alpha \in \{0, \cdots, k\}$ we have an integer ϵ_α (being 1 or 2) such that $u([t_\alpha, t_{\alpha+1}]) \subseteq X_{\epsilon_\alpha}$. By reparameterisation, we consider $u_{[t_\alpha, t_{\alpha+1}]}$ to be the same as a dimap $u_\alpha : \vec{I} \to X$. Now by definition of the composition in $\vec{\pi}_1(X)$ we have,

$$[u] = \vec{\pi}_1(i_{\epsilon_k})[u_k] \circ \ldots \circ \vec{\pi}_1(i_{\epsilon_0})[u_0]$$

Therefore the morphism f we are looking for, if it exists, must necessarily satisfy:

$$f[u] = f_{\epsilon_k}[u_k] \circ \ldots \circ f_{\epsilon_0}[u_0]$$

Let us define f this way and check that this is a correct definition.

First, we have to prove that this definition does not depend on the subdivision $0 = t_0 < t_1 < \cdots < t_{k+1} = 1$ that has been chosen.

Let $0 = t'_0 < t'_1 < \cdots < t'_{k'+1} = 1$ be another possible choice of subdivision. Consider the new one, $0 < t''_0 < t''_1 < \cdots < t''_{k+k'+2} = 1$ union of this subdivision with $0 < t_0 < \cdots < t_{k+1}$. So, t''_i is some t_j or t'_j for some j (result of the merge sort of t_* with the t'_*). This means that for some j_i and j_{i+1} we have $t''_{j_i} = t_i$ and $t''_{j_{i+1}} = t_{i+1}$. Let $u''_j : [t''_j, t''_{j+1}] \to X_{\epsilon''_j}$ be the corresponding restriction of dipath u. We now prove that $f_{\epsilon_j}(u_j) = f_{\epsilon''_{j_{i+1}-1}}(u''_{j_{i+1}-1}) \circ \cdots \circ f_{\epsilon''_{j_i}}(u''_{j_i})$, by identifying u''_k with their dihomotopic counterparts (by a reparameterization) $u''_k : [0,1] \to X_{\epsilon''_k}$. Notice that u''_k has its values in $X_{\epsilon''_k} \cap X_{\epsilon_0}$. But as $f \circ j_1 = f \circ j_2$, $f_{\epsilon_0}([u''_k]) = f_{\epsilon''_j}([u''_k])$ so $f_{\epsilon''_{j_{i+1}-1}}([u''_{j_{i+1}-1}]) \circ \cdots \circ f_{\epsilon''_{j_i}}([u''_{j_i}]) = f_{\epsilon_0}([u''_{j_{i+1}-1}]) \circ \cdots \circ f_{\epsilon_0}([u''_{j_i}])$ which is obviously equal to $f_{\epsilon_j}([u_j])$. This proves (by induction) that the definition of f does not depend on the subdivision.

Now we prove that $f([u])$ only depends on the class of u and not on u. Let $h : \vec{I} \times I \to X$ be a dihomotopy linking the dipath u to the dipath v. Let δ be the Lebesgue number of the covering of $\vec{I} \times I$ by $h^{-1}(\overset{\circ}{X}_1)$ and $h^{-1}(\overset{\circ}{X}_2)$. Consider subdivisions

$$\begin{array}{l} 0 = t_0 < \cdots < t_{k+1} = 1 \\ 0 = \tau_0 < \cdots < \tau_{l+1} = 1 \end{array}$$

so that all squares $[t_i, t_{i+1}] \times [\tau_j, \tau_{j+1}]$ have diameter less than δ. Further subdivide so that each $\tau \to h(t_i, \tau)$ for $\tau \in [\tau_j, \tau_{j+1}]$ is a dipath or an anti-dipath (a dipath with the reverse ordering on $\vec{I}$). This is possible since these are continuous paths which can be decomposed into a finite number of dipaths or anti-dipaths by hypothesis on X.

Let $u'(t) = h(t, \tau_1)$. If we prove $f([u]) = f([u'])$ then it follows by an easy induction that $f([u]) = f([v])$. Let P_α be the rectangle between lines $t = t_\alpha$, $t = t_{\alpha+1}$, $\tau = 0$ and $\tau = \tau_1$. Call,

$$\begin{array}{ll} u_\alpha : [t_\alpha, t_{\alpha+1}] & \to X_{\epsilon_\alpha} \\ u'_\alpha : [t_\alpha, t_{\alpha+1}] & \to X_{\epsilon_\alpha} \\ w_\alpha : [0, \tau_1] & \to X_{\epsilon_\alpha} \end{array}$$

the functions $u_\alpha(t) = h(t, 0)$, $u'_\alpha(t) = h(t, \tau_1)$ and $w_\alpha(\tau) = h(t_\alpha, \tau)$. We use the same names for any reparameterization of these functions from $[0, 1]$ to X_{ϵ_α}. Of course $w_0 = [w_0] = Id$ and $w_{k+1} = [w_{k+1}] = Id$.

We now notice the following in $\vec{\pi}_1(X)$ (since we have dihomotopies h on each P_α):

- Suppose w_α is a dipath,
  - suppose $w_{\alpha+1}$ is a dipath, then it is easy to see that $[u'_\alpha] \circ [w_\alpha] = [w_{\alpha+1}] \circ [u_\alpha]$ $\left(h'(x,t) = \begin{cases} h(0, 3x(1-t)) & \text{if } x \leqslant \frac{1}{3} \\ h(3x-1, 1-t) & \text{if } \frac{1}{3} \leqslant x \leqslant \frac{2}{3} \\ h(1, 3(x-1)t+1) & \text{if } \frac{2}{3} \leqslant x \leqslant 1 \end{cases} \right.$ is a dihomotopy between $u'_\alpha \circ w_\alpha$ and $w_{\alpha+1} \circ u_\alpha$ $\Big)$
  - suppose $w_{\alpha+1}$ is an anti-dipath, then $w_{\alpha+1}^{-1}$ defined as being $w_{\alpha+1}^{-1}(t) = w_{\alpha+1}(1-t)$, for $t \in [0,1]$, is a dipath and $[u'_\alpha] \circ [w_\alpha] = [w_{\alpha+1}^{-1}] \circ [u_\alpha]$. $\left(\text{consider dihomotopy } h'(x,t) = \begin{cases} h(0, 3xt)) & \text{if } x \leqslant \frac{1}{3} \\ h(3x-1, t) & \text{if } \frac{1}{3} \leqslant x \leqslant \frac{2}{3} \\ h(1, 3t(1-x)) & \text{if } \frac{2}{3} \leqslant x \leqslant 1 \end{cases} \right)$
- Similarly, suppose w_α is an anti-dipath,
  - suppose $w_{\alpha+1}$ is a dipath, then $[u_\alpha] \circ [w_\alpha^{-1}] = [w_{\alpha+1}] \circ [u'_\alpha]$.
  - suppose $w_{\alpha+1}$ is an anti-dipath, then $[u_\alpha] \circ [w_\alpha^{-1}] = [w_{\alpha+1}^{-1}] \circ [u'_\alpha]$.

Consider now:

$$\begin{array}{r} f([u]) = f_{\epsilon_k}([u_k]) \circ \cdots \circ f_{\epsilon_0}([u_0]) \\ f(([u']) = f_{\epsilon_k}([u'_k]) \circ \cdots \circ f_{\epsilon_0}([u'_0]) \end{array}$$

We also have,

$$f([u]) = f_{\epsilon_k}([w_{k+1}]) \circ f_{\epsilon_k}([u_k]) \circ \cdots \circ f_{\epsilon_0}([u_0])$$
$$f(([u']) = f_{\epsilon_k}([u'_k]) \circ \cdots \circ f_{\epsilon_0}([u'_0]) \circ f_{\epsilon_0}([w_0])$$

We prove by induction on α that,

$$f_{\epsilon_\alpha}([u'_{\alpha-1}]) \circ \cdots \circ f_{\epsilon_0}([u'_0]) \circ f_{\epsilon_0}([w_0]) = f_{\epsilon_\alpha}([w_\alpha^{j_\alpha}]) \circ f_{\epsilon_{\alpha-1}}([u_{\alpha-1}]) \circ \cdots \circ f_{\epsilon_0}([u_0])$$

with j_α being -1 if w_α is an anti-dipath, $+1$ otherwise.

This is a direct consequence of the remark above. □

In [**34**], M. Grandis proves a general Seifert/van Kampen theorem for a form of dihomotopy. In fact, in his case, dihomotopies are maps from $\vec{I} \times \vec{I} \to X$, so two dipaths are equivalent modulo dihomotopy if there is a sequence of such dihomotopies linking one with the other (giving these zig-zag dipaths as in the proof above). We believe that in the case of the geometric realization of finite (pre) cubical sets, which is then in particular compact, these two notions of fundamental categories coincide, and that the general version of Seifert/van Kampen holds.

In fact, the Seifert/van Kampen theorem works right away in the combinatorial case. In order to do this, we have to define an analogous to the fundamental category, the edge-path category. We sketch the construction in the easier case of precubical sets, but this can be done as well for cubical sets.

A dipath in a precubical set (M, d^0, d^1) is a finite sequence ($k \geqslant 1$)

$$p = (p_1, \cdots, p_k)$$

where all p_i are 1 dimensional such that $d^1_0(p_i) = d^0_0(p_{i+1})$ or the empty sequence $\emptyset$.

The initial state of a non-empty dipath p is $s(p) = d^0_0(p_1)$ and its final state is $t(p) = d^1_1(p_k)$.

The composition $*$ (or concatenation) of dipaths is as follows. Let $p = (p_1, \cdots, p_k)$ and let $q = (q_1, \cdots, q_l)$ be two non-empty dipaths such that the initial state of q is the final state of p. Then $p * q = (p_1, \cdots, p_k, q_1, \cdots, q_k)$ (is a dipath indeed).

This composition is associative has as neutral element the empty dipath. Let $[p]$ denote the contiguity class of dipath p. Concatenation induces an operation, still denoted by $*$, with $[p] * [q] = [p * q]$.

Define the edge-path category $\mathcal{E}(M)$ of M as follows. Its objects are elements of M_0. Its morphisms from $\alpha \in M_0$ to $\beta \in M_0$ are contiguity classes of dipaths from α to β, or the empty dipath. The composition between morphisms is $*$ (we write now $[q] \circ [p] = [p] * [q]$). This forms a category.

Let now $f : M \to N$ be a morphism of precubical sets. It is easy to see that f maps dipaths of M to dipaths of N and that if p and q are two contiguous dipaths in M, then $f(p)$ and $f(q)$ are two contiguous dipaths in N. Similarly, $f(p * q) = f(p) * f(q)$. Therefore f induces a transformation $\mathcal{E}(f)$ transforming objects of $\mathcal{E}(M)$ into objects of $\mathcal{E}(N)$, morphisms of $\mathcal{E}(M)$ into $\mathcal{E}(N)$, respecting composition. Hence $\mathcal{E}$ is a functor from the category of precubical sets to the category of categories.

Proposition 5. *Let $X = X_1 \cup X_2$ be a finite precubical set, union of two precubical sets. Call $j_1 : X_1 \cap X_2 \to X_1$ (respectively $j_2 : X_1 \cap X_2 \to X_2$) and $i_1 : X_1 \to X$*

(respectively $i_2 : X_2 \to X$) the natural inclusion morphisms. Then the following diagram

$$\begin{tikzcd} \mathcal{E}(X_1 \cap X_2) \arrow[r, "\mathcal{E}(j_1)"] \arrow[d, "\mathcal{E}(j_2)"'] & \mathcal{E}(X_1) \arrow[d, "\mathcal{E}(i_1)"] \\ \mathcal{E}(X_2) \arrow[r, "\mathcal{E}(i_2)"'] & \mathcal{E}(\mathcal{X}) \end{tikzcd}$$

is co-cartesian.

Proof. Same as above, but simpler in that, there is no Lebesgue number argument (replaced by the finiteness argument), and all un-directed paths in $X_1 \cap X_2$ are now always compositions of a finite number of di- and anti-di- paths (no achronal part!). □

9.2. Higher-order homotopies

We are going to define inductively the notion of higher-order dihomotopy. We call dihomotopy of order 0 between two point x and y any dipath from x to y. A dihomotopy of order 1 is any dihomotopy between two dipaths with the same ends. Now, suppose we have defined dihomotopies of order up to n ($n \geqslant 1$) between dihomotopies of order $n-1$ with end fixed (gathered in a set $\vec{P}_n(X)$):

Definition 17. *A dihomotopy of order $n+1$ ($n \geqslant 1$) between dihomotopies H, G of order n with equal ends is a dimap $A : \vec{I} \times I^{n+1} \to X$ such that for all $x \in \vec{I} \times I^n$, $A(x,0) = H(x)$ and $A(x,1) = G(x)$. The source of A is $s_n(A) = H$ and its target is $t_n(A) = G$. The set of such dihomotopies is noted $\vec{P}_n(X)$.*

We can define compositions on the sets $\vec{P}_n(X)$ ($n \geqslant 1$) as follows; $*_{n-1} : \vec{P}_n(X) \times \vec{P}_n(X) \to \vec{P}_n(X)$ is defined for (f,g) such that $t_n(f) = s_n(g)$:

$$(f *_{n-1} g)(x_0, \cdots, x_n) = \left\{ \begin{array}{ll} 0 \leqslant x_n \leqslant \frac{1}{2} & f(x_0, \cdots, x_{n-1}, 2x_n) \\ \frac{1}{2} \leqslant x_n \leqslant 1 & g(x_0, \cdots, x_{n-1}, 2x_n - 1) \end{array} \right.$$

This naturally gives groupoïds π_n for all dihomotopies of higher-dimension $n \geqslant 2$ modulo dihomotopies of dimension $n+1$. Of course, if we take a base path and look at loops around this base path we have π_2 becoming a group, and π_n ($n \geqslant 3$) becoming abelian groups.

The question one has to solve now is: do we have exact sequences such as the homotopy exact sequence of a pair? Do we have interesting spectral sequences? Does all this come from a closed-model structure?

10. Conclusion and Future Work

The dihomotopy theory and applications to concurrency gradually developed together, but there is much left to do as I tried to show throughout this text.

We would like to consider also other potential applications. In fact, there are a certain number of other "geometric" theories which apply to computation models. For instance, there are algebraic topological considerations in linear logics (see for instance [**48**], [**49**], [**2**] and [**3**]) and in modal logics [**32**] which might be related to the subject of this paper. More generally, linear logic [**24**] has a strong geometric flavour and can be understood as a logic dealing with resources. There are also very nice results in sequential computation relating topological invariants with computability results, such as Squier's theorem, see [**66**] for instance, and also some very important applications of geometrical ideas to distributed computing, see for instance [**36**]. I believe that some fruitful cross-fertilizations should bring exciting new results in the years to come.

References

[**1**] Arnold, A., "Systèmes de transitions finis et sémantique des processus communicants," Masson, 1992.

[**2**] Basu, K., *The geometry of sequential computation I: A simplicial geometry of interaction*, Technical report, Institutsbericht, Technische Universitaet Muenchen, Institut fuer Informatik (1997).

[**3**] Basu, K., *The geometry of sequential computation II: A simplicial geometry of interaction*, Technical report, Institutsbericht, Technische Universitaet Muenchen, Institut fuer Informatik (1997).

[**4**] Baues, H. J., *Foundations of proper homotopy theory* (1992).

[**5**] Bednarczyk, M. A., "Categories of asynchronous systems," Ph.D. thesis, University of Sussex (1988).

[**6**] Brown, R. and P. J. Higgins, *Colimit theorems for relative homotopy groups*, Journal of Pure and Applied Algebra (1981), pp. 11–41.

[**7**] Brown, R. and P. J. Higgins, *On the algebra of cubes*, Journal of Pure and Applied Algebra (1981), pp. 233–260.

[**8**] Buckland, R. and M. Johnson, *ECHIDNA: A system for manipulating explicit choice higher dimensional automata*, in: *AMAST'96: Fifth Int. Conf. on Algebraic Methodology and Software Technology*, Munich, 1996.

[**9**] Coffman, E. G., M. J. Elphick and A. Shoshani, *System deadlocks*, Computing Surveys **3** (1971), pp. 67–78.

[**10**] Dijkstra, E., "Cooperating Sequential Processes," Academic Press, 1968.

[**11**] Escardo, M., *The regular-locally-compact coreflection of a stably locally compact locale*, Journal of Pure and Applied Algebra **157** (2001), pp. 41–55.

[**12**] Fahd, A. A. A.-A., R. Brown and R. Steiner, *Multiple categories: the equivalence of a globular and a cubical approach*, Technical report, arXiv:math.CT/0007009 (2000).

[**13**] Fajstrup, L., *Loops, ditopology, and deadlocks*, Mathematical Structures in Computer Science **10(4)** (2000).

[14] Fajstrup, L., E. Goubault and M. Raussen, *Detecting deadlocks in concurrent systems*, in: *Proceedings of the 9th International Conference on Concurrency Theory, also available at http://www.dmi.ens.fr/~goubault* (1998).

[15] Fajstrup, L., E. Goubault and M. Raussen, *Algebraic topology and concurrency*, submitted to Theoretical Computer Science, also technical report, Aalborg University (1999).

[16] Fajstrup, L. and S. Sokolowski, *Infinitely running processes with loops from a geometric view-point*, Electronic Notes in Theoretical Computer Science, Proceedings of GETCO'00 (2000).

[17] Freyd, P. and A. Scedrov, "Categories, Allegories," volume 39 in Mathematical Library, North-Holland, 1990.

[18] Gabriel, P. and M. Zisman, *Calculus of fractions and homotopy theory*Number 35 in Ergebnisse der Mathematik und ihrer Grenzgebiete, Springer Verlag, 1967 .

[19] Gaucher, P., *About the globular homology of higher dimensional automata* (2000), preprint available at math.CT/0002216.

[20] Gaucher, P., *Combinatorics of branchings in higher dimensional automata* (2000), preprint available at math.CT/9912059.

[21] Gaucher, P., *From concurrency to algebraic topology*, Electronic Notes in Computer Science **39** (2000).

[22] Gaucher, P., *Homotopy invariants of higher dimensional categories and concurrency in computer science*, Mathematical Structures in Computer Science **10(4)** (2000).

[23] Gaucher, P. and E. Goubault, *Topological deformation of higher dimensional automata*, Technical report, arXiv:math.AT/010760 (2001).

[24] Girard, J.-Y., *Linear logic, its syntax and semantics*Number 222 in Advances in Linear Logic, Cambridge University Press, London Mathematical Society Lecture Notes Series, 1995 .

[25] Goubault, E., *Domains of higher-dimensional automata*, in: *Proc. of CONCUR'93* (1993).

[26] Goubault, E., "The Geometry of Concurrency," Ph.D. thesis, Ecole Normale Supérieure (1995), also available at http://www.dmi.ens.fr/~goubault.

[27] Goubault, E., *Schedulers as abstract interpretations of HDA*, in: *Proc. of PEPM'95* (1995).

[28] Goubault, E., *Geometry and concurrency: A users' guide*, Mathematical Structures in Computer Science (2000).

[29] Goubault, E., *Cubical sets are generalized transition systems*, Technical report, submitted, also available at http://www.di.ens.fr/~goubault (2001).

[30] Goubault, E. and T. P. Jensen, *Homology of higher-dimensional automata*, in: *Proc. of CONCUR'92* (1992).

[31] Goubault, E. and M. Raussen, *Dihomotopy as a tool in state space analysis*, in: *Proceedings of LATIN'02*, 2002.

[32] Goubault-Larrecq, J. and E. Goubault, *Order-theoretic, geometric and combinatorial models of intuitionistic S4 proofs*, in: *IMLA'99*, 1999.

[33] Grandis, M., *An intrinsic homotopy theory for simplicial complexes, with applications to image analysis*, Technical report, Appl. Categ. Structures, to appear. Preliminary version: Dip. Mat. Univ. Genova, Preprint 383. Also at: http://arXiv.org/abs/math.AT/0009166 (1999).

[34] Grandis, M., *Directed homotopy theory, i. the fundamental category*, Cahiers Top. Gom. Diff. Catg, to appear, Preliminary version: Dip. Mat. Univ. Genova, Preprint 443 (2001).

[35] Gunawardena, J., *Homotopy and concurrency*, in: *Bulletin of the EATCS*, 54, 1994, pp. 184–193.

[36] Herlihy, M. and S. Rajsbaum, *A primer on algebraic topology and distributed computing*, in: *Lecture Notes in Computer Science*, 1000 (2000).

[37] Hoare, C., "Communicating Sequential Processes," Prentice-Hall, 1985.

[38] Johnstone, P. T., "Stone Spaces," Cambridge University Press, 1982.

[39] Kegelman, M., "Continuous Domains in Logical Form," Ph.D. thesis (1999).

[40] Kopperman, R., *Communication at the Dagstuhl seminar 00231: Topology in computer science: Constructivity; asymmetry and partiality; digitization* (2000).

[41] Lundell, A. and S. Weingram, "The topology of CW-complexes," Van Nostrand Reinhold, 1969.

[42] Lynch, N., "Distributed Algorithms," Morgan-Kaufmann, 1996.

[43] Mac Lane, S., "Categories for the working mathematician," Springer-Verlag, 1971.

[44] Maclane, S. and I. Moerdijk, "Sheaves in Geometry and Logic," Springer-Verlag, 1992.

[45] May, J. P., "Simplicial objects in algebraic topology," D. van Nostrand Company, 1967.

[46] Mazurkiewicz, A., *Basic notions of trace theory*, in: *Lecture notes for the REX summer school in temporal logic* (1988).

[47] Mc Cleary, J., "User's guide to spectral sequences," Publish or perish, Mathematics lecture series 12, 1985.

[48] Métayer, F., *Homology of proof-nets*, Archive of Mathematical Logic **33** (1994), pp. 169–188.

[49] Métayer, F., *Volume of multiplicative formulas and provability*in: J.-Y. Girard, Y. Lafont and L. Regnier, editors, *Advances in Linear Logic*, Cambridge University Press, 1995 pp. 297–306, proceedings of the Workshop on Linear Logic, Ithaca, New York, June 1993.

[50] Milner, R., "Calculus of Communicating System," Number 92 in LNCS, Springer-Verlag, 1980.

[51] Milner, R., "Communication and Concurrency," Prentice Hall, 1989.

[52] Nachbin, L., "Topology and Order," Van Nostrand, Princeton, 1965.

[53] Penrose, R., "Techniques of Differential Topology in Relativity," Conference Board of the Mathematical Sciences, Regional Conference Series in Applied Ma thematics **7**, SIAM, Philadelphia, USA, 1972.

[54] Pratt, V., *Modeling concurrency with geometry*, in: *Proc. of the 18th ACM Symposium on Principles of Programming Languages* (1991).

[55] Pratt, V., *Chu spaces: Automata with quantum aspects*, Technical report, Stanford University (1994).

[56] Raussen, M., *On the classification of dipaths in geometric models for concurrency*, Mathematical Structures in Computer Science **10(4)** (2000).

[57] Raussen, M., *Topological antidotes to the state space explosion problem*, Technical Report R-01-2021, Dept. of Mathematics, Aalborg University, Aalborg, Denmark (2001).

[58] Serre, J., "Homologie Singulière des Espaces Fibrés. Applications," Ph.D. thesis, Ecole Normale Supérieure (1951).

[59] Shields, M., *Concurrent machines*, Computer Journal **28** (1985).

[60] Shoshani, A. and E. G. Coffman, *Sequencing tasks in multiprocess systems to avoid deadlocks*, in: *Conference Record of 1970 Eleventh Annual Symposium on Switching and Automata Theory*, IEEE, Santa Monica, California, 1970, pp. 225–235.

[61] Sokolowski, S., *Homotopy in concurrent processes*, Technical report, Institute of Computer Science, Gdansk Division (1998).

[62] Sokolowski, S., *Investigation of concurrent processes by means of homotopy functors*, Technical report, Institute of Computer Science, Gdansk Division (1998).

[63] Sokolowski, S., *Point glueing in cpo-s*, Technical report, Institute of Computer Science, Gdansk Division (1998).

[64] Sokolowski, S., *Classifying holes of arbitrary dimensions in partially ordered cubes*, Technical report, Personal Communication (1999).

[65] Spanier, E. J., "Algebraic Topology," McGraw Hill, 1966.

[66] Squier, C., *Word problems and a homological finiteness condition for monoids*, J. of Pure and Applied Algebra **49** (1987), pp. 201–217.

[67] Street, R., *The algebra of oriented simplexes*, J. Pure Appl. Algebra (1987), pp. 283–335.

[68] Winskel, G. and M. Nielsen, "Models for concurrency," 3, Oxford University Press, 1994 .

This article may be accessed via WWW at http://www.rmi.acnet.ge/hha/
or by anonymous ftp at
ftp://ftp.rmi.acnet.ge/pub/hha/volumes/2003/n2a5/v5n2a5.(dvi,ps,pdf)

Eric Goubault `Eric.Goubault@cea.fr`

DTSI/SLA,
CEA/Saclay,
91191 Gif-sur-Yvette, FRANCE