Chemistry in Action:

Discovering the Behaviour of a Network from Local Observations

Stefan Heymer

Jens Grabowski

October 6, 1998
Report A-98-18

Abstract

Chemical abstract machines (CHAMs) are used as the model to solve the problem of finding possible communication paths between network nodes. The information used for the construction of the CHAMs is retrieved by performing local observations at the different network nodes. We introduce the model of CHAMs, explain the construction of different types of CHAMs to solve our problem, and show the applicability of our approach by describing a TCP/IP based experiment.

CR Categories and Subject Descriptors:
F.1.1 [Theory of Computation]: Models of Computation; C.2.3 [Computer Communication Networks]: Network Management; C.2.2 [Computer Communication Networks]: Network Protocols

General Terms:
Chemical Abstract Machines, Discovery of Network Behaviour, TCP/IP

Chemistry in Action: Discovering the Behaviour of a Network from Local Observations

Stefan Heymer and Jens Grabowski

Institute for Telematics, Medical University of Lübeck, Ratzeburger Allee 160,
D–23538 Lübeck, Germany
phone +49 451 500 3724, fax +49 451 500 3722
{heymer,grabowsk}@itm.mu-luebeck.de

1 Motivation

Networks often grow in an evolutionary fashion. Nodes are attached to or deleted from a network mostly without considering the effect of these modifications onto the network behaviour as a whole. For instance, this is the case in TCP/IP networks, but as well for telephony networks, where the network is planned far more thoroughly.

In TCP/IP networks as well as in telephony networks, the operator does not have a clear understanding of the route which a connection or communication takes through the network. In most networks, an operator is only able observe the load of the different nodes and some characteristics of the traffic between them. The concrete information flow of a connection, i.e. the channels and gateways used, and the order of their use cannot be observed.

Software support to tackle this problem is available for local area networks (LANs) and TCP/IP networks with SNMP software. The situation is worse for non–TCP/IP networks, like telephony networks. Only local probes of the communication and signalling traffic at the network nodes are available. Such local probes are often also available for LANs and TCP/IP networks.

Our aim is to define a model for the communication behaviour of a network which can be constructed from the local observations of the network nodes. As you will see in the examples in this paper, additional information can be gained from the combination of the local views, e.g. it is possible to detect additional communication paths which have not been observed locally. In case of network errors, such information may be valuable for network operators. However, until now we have restricted our work to a path finding problem, but we hope that our work can also be used to predict the communication behaviour of a network in case of addition or deletion of network nodes. For such problems our network model may need simulation facilities.

In this paper, we will give an algorithm to combine local observations into a *global* model of the communication behaviour of the network. For this, we make use of two approaches based on ideas coming from chemistry and biology. Both approaches are rarely employed in computer science. We will use the chemical computation metaphor presented in [6] as our formalism of choice, and will use the idea of molecular computation presented in [1] as an inspiration to our solution. The biological and chemical metaphors have been used for two reasons: On the one hand, routing in LANs and telephony networks is performed in a massively parallel way. Hence, we are looking for formalisms capturing this aspect. On the other hand, the formalism of CHAMs introduced below allows the transformation into models of process systems, i.e. simulation models. Our solution given in Section 3 could also be described using concepts of graph theory only, but then there would be no formally clean way to obtain simulation models.

The rest of the paper is organised in the following manner: Chemical Abstract Machines and DNA computing are introduced in Section 2. The construction of different Chemical Abstract Machines from local observations is described and discussed in the main part of this paper (Section 3). A case study showing the application of our approach is described in Section 4. In Section 5 we conclude and give an outlook on future work.

2 Preliminaries

Before presenting a solution to the problem of discovering network behaviour from looking just at local observations, we introduce the notion of Chemical Abstract Machines and the idea of DNA computing.

2.1 Chemical Abstract Machines

The model of Chemical Abstract Machines (CHAMs) is based on the chemical metaphor first put forward by Banâtre and Le Métayer in [2, 3]. The idea behind this metaphor is to provide a model where concurrent components are "moving" freely in a system, communicating with each other when coming into contact. The metaphor is realised in the Gamma formalism.

Intuitively, a system state in Gamma is like a *chemical solution*, where molecules in the solution are able to react which each other according to *reaction rules*. The solution is stirred by a *magical mechanism*, providing possible contacts between molecules. This magical mechanism is an "implementation" of the usual Brownian motion found in chemical solutions.

Formally, a chemical solution can simply be seen as a finite multiset of elements (which are called *molecules*), denoted by

$$S = \{\!|m_1, m_2, \ldots, m_k|\!\}.$$

This also accounts for the stirring mechanism, as the elements are unordered and may be assumed to make contact arbitrarily.

The notion of a CHAM was introduced in [5] and developed further in [6]. The Gamma formalism is enhanced by extending the use of multisets, allowing them to be part of the molecules, and by introducing a classification of transformation rules. The CHAM is meant to be a framework for the description of the operational semantics of parallel languages and for the examination and development of parallel computations.

A CHAM consists of a set of molecules $m, m', \ldots$ and a state transition relation $S \to S'$, with the states $S, S', \ldots$ being finite multisets of molecules, called *solutions*. At the abstract level, we assume the molecules to be terms built according to some given syntax.

Also, the standard notion of *context* as being terms with open places in them is used. The term $C[T]$ denotes the term that one gets when filling the open place in the context $C[\cdot]$ with the term T. A general membrane construct is used to transform a solution into a single molecule. For this construct, no special notation is used, meaning that for any CHAM, any solution is assumed to be a molecule as well.

Solution transformations $S \to S'$ are given by means of rules. There are just two general rules, called "laws", which hold for any CHAM. The first law, the *Chemical Law*, states that reactions

can be performed freely in any solution ($\uplus$ denoting multiset union):

$$\frac{S \to S'}{S \uplus S'' \to S' \uplus S''} \tag{1}$$

The second law, the *Membrane Law*, asserts that solutions can evolve freely in any molecule context:

$$\frac{S \to S'}{\{|C[S]|\} \to \{|C[S']|\}} \tag{2}$$

These two rules are the only rules that involve an induction on the behaviour of a solution S: Every other rule is purely local, i.e. it will only concern the molecules participating in a reaction.

In [7], the reaction rules $S \to S'$ are classified into two kinds, formalising the transition relation $\to$ as two binary relations over multisets $\rightleftharpoons$ and $\mapsto$. Intuitively, transitions $S \mapsto S'$ represent *proper reactions* changing the solution on the left–hand side *permanently* to the solution on the right–hand side. The reacting molecules may be called *ions*, as they often also exhibit interaction capabilities (which were called *valences* in [6]) by having a special shape.

The other kind of transitions $S \rightleftharpoons S'$ are structural transformations and are not counted as evaluation steps. We say that a solution is *inert* if it cannot perform any reaction except for structural transformations. Structural transformations are reversible, like heating a chemical solution and cooling it down to the same solution again. In the scope of this paper, we will only consider reactions, but we will hint at the role structural transitions will play in our future work.

2.2 DNA computing

The deoxyribonucleic acid (DNA) encodes genes and most of the operations in biological cells. It uses a four letter alphabet of bases: `A` (adenine), `C` (cytosine), `G` (guanine) and `T` (thymine). When viewed as a long string of letters, a strand of DNA does resemble the tape of a Turing machine. Yet, from a molecular and chemical point of view, many operations on such strings are nontrivial.

DNA normally comes double–stranded, containing two long strings of bases twisted around each other in a helical form. The two strands are held together by specific binding forces between the bases: A binds to T, and C binds to G, and vice versa. Hence, the sequence of bases in one strand is complementary to the sequence of bases in the other strand, forming its Watson–Crick complement. For example, the strand `ACTGAG` lines up with its Watson–Crick complement `TGACTC` to form the following double–stranded molecule:

```
ACTGAG
TGACTC
```

In biological systems, DNA is reproduced by copying. Enzymes, specialised macro molecules, are able to match patterns in double stranded DNA and to "cut" through the strands. For instance, some enzyme may cut the sequence

```
GAATTC
CTTAAG
```

into the following subsequences:

```
G         AATTC
CTTAA       G
```

Note that the enzyme does not cut straight through the DNA: it leaves two short ends of single–stranded DNA. These "sticky" ends may later realign either with their original partner, or with some other complementary end floating around in the solution. Recombinant DNA technology is based on these operations: cut DNA and permit single–stranded ends to find new partners and recombine.

Adleman reported in [1] on how to solve a 7–node Hamiltonian path problem with these techniques. We will show here a solution by Beaver (taken from [4]), which will provide a starting point for our initial problem.

Given a graph $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ being the vertices, v_1 being the source and v_n being the destination, several small molecules are constructed representing the edges in E. Each edge will be given a number of different representations, one for each position which the edge can take in a complete path. This also restricts the length of possible paths, infinite paths cannot be generated. Hence, a molecule $e(v_i, v_j; d)$ is constructed to represent the edge $(v_i, v_j) \in E$ as the d^{th} edge in a possible path.

For this, n^2 molecules $v(v_i, d)$ are constructed to encode the vertices v_i from V at the respective positions d on the path, together with their n^2 Watson–Crick complements $\overline{v}(v_i, d)$. Beaver gives in [4] for $n = 4$ the following example coding with five bases:

i	$v(v_i,1)$	$v(v_i,2)$	$v(v_i,3)$	$v(v_i,4)$	$\overline{v}(v_i,1)$	$\overline{v}(v_i,2)$	$\overline{v}(v_i,3)$	$\overline{v}(v_i,4)$
1	AACGA	AACGC	AACGG	AACGT	TTGCT	TTGCG	TTGCC	TTGCA
2	CACGA	CACGC	CACGG	CACGT	GTGCT	GTGCG	GTGCC	GTGCA
3	GACGA	GACGC	GACGG	GACGT	CTGCT	CTGCG	CTGCC	CTGCA
4	TACGA	TACGC	TACGG	TACGT	ATGCT	ATGCG	ATGCC	ATGCA

Molecules $e(v_i, v_j; d)$ then are assembled from two sequences $\overline{v}(v_i, d)$ and $v(v_j, d+1)$, glued together by an arbitrary, but fixed intervening sequence:

$$\begin{array}{ll} \overline{v}(v_i, d) & \texttt{GGGGG} \\ & \texttt{CCCCC} \quad v(v_j, d+1) \end{array}$$

Hence, molecules $e(v_i, v_j; d)$ and $e(v_j, v_k; d+1)$ can combine with build a larger molecule, since the single–stranded right end of $e(v_i, v_j; d)$ is complementary to the single–stranded left end of $e(v_j, v_k; d+1)$. In general, $e(v_i, v_j; d)$ can combine with any one of several candidates of the form $e(j, k'; d+1)$. Thus, the possibility of pairing with different edges permits various paths to be generated at random, depending on the random order in which the molecules collide. As more than one molecule is generated for each edge, multiple paths can be examined in parallel.

For finding a Hamiltonian path in the graph, two special sequences LP and RP are generated first. These sequences work as primers for a process called polymerase chain reaction (PCR): molecules containing these primers get duplicated in the PCR, providing an emphasis of these molecules in the test tubes. Molecules for the start L and end R of a path are constructed from these primers in the following way:

$$\begin{array}{ll} \overbrace{\overline{LP}}^{L} & \overbrace{\overline{v}(n,n) \quad \overline{RP}}^{R} \\ LP \quad v(1,1) & RP \end{array}$$

To solve the problem of finding a Hamiltonian path in a graph using this method, for each edge $(v_i, v_j) \in E$ and for each d with $1 \leq d \leq n-1$ molecules $e(v_i, v_j; d)$ are synthesised, the molecules L and R are synthesised, the various fragments are mixed and allowed to anneal. As the vertices are numbered with step numbers, chains found in the solution may have a length of at most $n-1$ edge sequences plus the length of the sequences for the L and R molecules. PCR is then used to amplify the sequences which contain the primer sequences. Solutions of the problem are those molecules which contain both primer sequences, provided the graph is free of loops.

3 Discovering the Behaviour of a Network

Our problem is to find a suitable model, i.e. a CHAM, for some aspects of network behaviour which can be constructed from local observations of the network nodes. The aspect of the network behaviour which we want to analyse is the possible information flow between terminal nodes. In other words, we are looking for communication paths between telephones, fax devices, or PCs.

We assume that some nodes in the network are able to initiate tests by trying to build up connections to other nodes. The results for these tests can be either observed by the testing node itself, as it is the case with the UNIX `ping` and `traceroute` commands, or by observing the input/output behaviour of nodes, e.g. gateways, on the route. Without taking into account this difference, we call these observations *local observations*, as they are made locally at a specific node.

However, we do not want to see a collection of local observations. Instead, we want to see how the network behaves *globally*. Hence, we have to condensate the information gained from the local observations into a *global observation*.

3.1 Formulating a CHAM

Now that we have given a description of the problem, we formulate some chemical abstract machines to solve this problem. To do this, we adapt some of the ideas presented for the Hamiltonian path problem in [4] to our setting. In our problem statement, we focus on the observation of paths

$$v_1 \to v_2 \to \ldots \to v_n$$

in a network with $v_1, v_2, \ldots, v_n$ being nodes of the network. To encode the directed nature of such a path, we use symbols $v(v_i, v_j)$ corresponding to the molecules $v(v_i, n)$ in Beaver's approach. Instead of encoding the position of a node in a path, we encode the destination of the path, as we do not want to restrict the nodes to appear at specific positions in discovered paths.

The small strands of DNA Beaver used in his approach can be encoded as pairs $\overline{v}(v_i, v_k)\underline{v}(v_j, v_k)$, denoting an edge from node v_i to node v_j on a path to v_k. As we use strings of symbols as our CHAM molecules, we do not need the intervening sequences to glue the single stranded DNA molecules, but mark the position on the "upper" or "lower" strand of the DNA molecule explicitly by using $\overline{v}(v_i, v_k)$ for molecules on the upper strand and $\underline{v}(v_j, v_k)$ for molecules on the lower strand. Pairs of such modules will be denoted with $v(v_i, v_k)$.

Hence, we are able to define the set of molecules used in our first two CHAMs.

Definition 1 *Let $V = \{v_1, v_2, \ldots, v_n\}$ be the set of nodes in the network to be examined. The molecules of the CHAMs $\mathcal{A}$ and $\mathcal{B}$ are taken from the set*

$$\begin{aligned} M \quad = \quad & \{\overline{v}(v_i, v_j)w \mid v_i, v_j \in V; w \in M'\} \\ \cup \quad & \{w\underline{v}(v_i, v_j) \mid v_i, v_j \in V; w \in M'\} \\ \cup \quad & \{\overline{v}(v_i, v_j)w\underline{v}(v_k, v_l) \mid v_i, v_j, v_k, v_l \in V; w \in M'\} \\ \cup \quad & M', \end{aligned}$$

with the set of molecules M' being defined as

$$M' = \{v(v_i, v_j) \mid v_i, v_j \in V\}^*.$$

To model the reactions in the test tube, we give just one special reaction rule for our first CHAM, the CHAM $\mathcal{A}$.

Definition 2 *Let the set of molecules M be defined as above. The special reaction rule for the CHAM $\mathcal{A}$ has the form*

$$u\underline{v}(v_i, v_j),\ \overline{v}(v_i, v_j)w \mapsto u\underline{v}(v_i, v_j),\ uv(v_i, v_j)w,\ \overline{v}(v_i, v_j)w.$$

This reaction rule does not "consume" its educts to model the sufficient supply of DNA strands in the test tubes. With this simple CHAM, we are able to formalise our problem. For an observed path

$$P = v_1 \to v_2 \to \ldots \to v_n$$

we encode the solution

$$enc(P) = \{|\overline{v}(v_1, v_n)\underline{v}(v_2, v_n), \ldots, \overline{v}(v_{n-1}, v_n)\underline{v}(v_n, v_n)|\}.$$

For a local observation $O = \{P_1, P_2, \ldots, P_m\}$, we thus are able to encode a *local solution* as

$$enc(O) = \uplus_{P \in O} enc(P),$$

joining the encoded solutions for the paths in an observation.

Example 1 *Let the set of nodes in a network be $V = \{a, b, c, d\}$, and assume that the local observation $O_a = \{a \to b, a \to c, a \to b \to d\}$ has been made at node a. The local solution for node a then is*

$$\begin{aligned} enc(O_a) \quad = \quad & \{|\overline{v}(a, b)\underline{v}(b, b)|\} \\ \uplus \quad & \{|\overline{v}(a, c)\underline{v}(c, c)|\} \\ \uplus \quad & \{|\overline{v}(a, d)\underline{v}(b, d), \overline{v}(b, d)\underline{v}(d, d)|\} \\ = \quad & \{|\overline{v}(a, b)\underline{v}(b, b), \overline{v}(a, c)\underline{v}(c, c), \overline{v}(a, d)\underline{v}(b, d), \overline{v}(b, d)\underline{v}(d, d)|\}. \end{aligned}$$

These local observations for each node can be combined into a *global solution* modelling the routing behaviour of the complete network. Again, this is done by simply joining the multisets of the solutions. Thus, let $O_{v_1}, O_{v_2}, \ldots O_{v_n}$ be the observations made for a network N. The encoding of N is

$$enc(N) = enc(O_{v_1}) \uplus enc(O_{v_2}) \uplus \ldots \uplus enc(O_{v_n}).$$

Example 2 *Let V be the set of nodes in the network N and O_a be the local observations for node a as given in Example 1. Assume further that for the node b the single path $b \to c \to d$ was observed, while for the nodes c and d no observations have been made. Hence, every node in the network is able to actively probe routes to other nodes and observe the results, but is not able to see routes from other nodes routed over itself.*

Then, the global solution for the network is determined as

$$\begin{array}{rcl} enc(N) & = & enc(O_a) \uplus enc(O_b) \uplus enc(O_c) \uplus enc(O_d) \\ & = & \{\!|\overline{v}(a,b)\underline{v}(b,b), \overline{v}(a,c)\underline{v}(c,c), \overline{v}(a,d)\underline{v}(b,d), \overline{v}(b,d)\underline{v}(d,d)|\!\} \\ & \uplus & \{\!|\overline{v}(b,d)\underline{v}(c,d), \overline{v}(c,d)\underline{v}(d,d)|\!\} \\ & \uplus & \{\!|\,|\!\} \uplus \{\!|\,|\!\} \\ & = & \{\!|\overline{v}(a,b)\underline{v}(b,b), \overline{v}(a,c)\underline{v}(c,c), \overline{v}(a,d)\underline{v}(b,d), \overline{v}(b,d)\underline{v}(d,d), \\ & & \overline{v}(b,d)\underline{v}(c,d), \overline{v}(c,d)\underline{v}(d,d)|\!\}. \end{array}$$

To find the routes between two nodes v_s and v_t in a network, we simply build the solution

$$\{\!|\underline{v}(v_s,v_t), \overline{v}(v_t,v_t)|\!\}$$

corresponding to the special molecules L and R in Beaver's solution, mix it into the global solution of the network and let reactions take place.

Example 3 *Let the encoding of the network be given as in Example 2. To compute some of the routes between the nodes b and d, we follow the reaction*

$$\begin{array}{rl} & enc(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ \mapsto & enc(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)\underline{v}(d,d), v(b,d)\underline{v}(c,d)|\!\} \\ \mapsto & enc(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)\underline{v}(d,d), v(b,d)\underline{v}(c,d)|\!\} \\ & \uplus \{\!|v(b,d)v(d,d), v(b,d)v(c,d)\underline{v}(d,d)|\!\} \quad , \\ \mapsto & enc(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)\underline{v}(d,d), v(b,d)\underline{v}(c,d)|\!\} \\ & \uplus \{\!|v(b,d)v(d,d), v(b,d)v(c,d)\underline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)v(c,d)v(d,d)|\!\} \end{array}$$

using in each step the special reaction rule from Definition 2. The result of the reactions is a solution containing the two inert molecules $v(b,d)v(d,d)$ and $v(b,d)v(c,d)v(d,d)$, which correspond to the paths $b \to d$ and $b \to c \to d$, respectively.

This last example shows that in the global solution, the whole is more than just the sum of its parts: the path $b \to d$ found in the reaction has not been given as an observation, but has been *discovered* in the reaction. Hence, when combining the local solutions into a global solution, possible interactions between paths encoded in different solutions can be inferred during the process of reaction.

But Example 3 also shows a problem of our first CHAM: the solutions obtained during the reactions become more and more unwieldy. This can easily be alleviated by using a variant reaction rule of the special reaction rule given in Definition 1, as is done in the definition of our second CHAM $\mathcal{B}$. This would consume parts of the solution, making the resulting solutions smaller.

Definition 3 *Let the set of molecules M be defined as in Definition 1. The special reaction rule for the CHAM $\mathcal{B}$ has the form*

$$u\underline{v}(v_i, v_j), \overline{v}(v_i, v_j)w \mapsto uv(v_i, v_j)w, \overline{v}(v_i, v_j)w.$$

This reaction rule forces the process of route discovery to terminate; yet it shifts the nondeterminacy of the search process from inside the reactions to the outside, letting the CHAM "decide" which way to choose.

But how does this new CHAM relate to our original CHAM $\mathcal{A}$? To state the formal connection, we have to define a normal form on reactions first.

Definition 4 *A reaction in the CHAM $\mathcal{A}$ is said to be in left–to–right form if and only if in each application of the reaction rule*

$$u\underline{v}(v_i, v_k), \overline{v}(v_i, v_k)w \mapsto u\underline{v}(v_i, v_k), uv(v_i, v_k)w, \overline{v}(v_i, v_k)w$$

we have $w = \underline{v}(v_j, v_k)$ for some node v_j of the network.

That is, a reaction is in left–to–right form if the molecules are built from left to right in a stepwise fashion, just adding single steps to the right end of a molecule. With this definition, we are able to state a normal form theorem.

Theorem 1 *Let N be a network, containing nodes v_s and v_t. Let S be a solution found in a reaction*

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S.$$

Then there exists a unique reaction in left–to–right form leading to the same solution S.
Proof: Straightforward. □

With this theorem, we are able to state the formal connection between CHAM $\mathcal{A}$ and CHAM $\mathcal{B}$. We begin with the observation, that every solution found by CHAM $\mathcal{B}$ can also be found by CHAM $\mathcal{A}$.

Theorem 2 *Let N be a network, containing nodes v_s and v_t. Let S be a solution found in a reaction*

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

containing a molecule m of the form

$$m = v(v_1, v_n)v(v_2, v_n) \ldots v(v_{n-1}, v_n)v(v_n, v_n),$$

in CHAM $\mathcal{B}$. Then there exists a unique reaction in CHAM $\mathcal{A}$ leading to a solution S' containing m.
Proof: In each step of the reaction

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S,$$

add the educt $uv(v_i, v_k)$ in the reaction step to the products of the step. The resulting chain of reactions will be a reaction inside CHAM $\mathcal{A}$. □

The main difference between CHAM $\mathcal{A}$ and CHAM $\mathcal{B}$ lies in the fact that CHAM $\mathcal{A}$ will compute *all* possible answers, given enough time, while CHAM $\mathcal{B}$ is only able to compute *one* specific answer. This is accounted for with the following theorem.

Theorem 3 *Let N be a network, containing nodes v_s and v_t. For each reaction*

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

performed by CHAM $\mathcal{A}$ containing an inert molecule m of the form

$$m = v(v_1, v_n)v(v_2, v_n) \ldots v(v_{n-1}, v_n)v(v_n, v_n),$$

we are able to find a reaction

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S'$$

of CHAM $\mathcal{B}$ with m being an element of S'.
Proof: Filter out of the chain of reactions

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

all those reactions not leading to educts used in the production of m. Bring the remaining reaction leading to m into left–to–right form, and remove the educts $u\underline{v}(v_i, v_j)$ from each reaction step. The resulting chain of reactions is a reaction inside CHAM $\mathcal{B}$. □

These two theorems establish that the CHAMs $\mathcal{A}$ and $\mathcal{B}$ in fact do find the same solutions. But the two CHAMs share a further problem: the local solutions encoding the observations are not inert — they are able to react inside themselves. This is alright and wanted when viewed from the standpoint of DNA computing, but complicating a possible implementation of our CHAMs. We will address this problem in the definition of our next CHAM.

3.2 Refined versions of the CHAMs

To overcome both problems, the growing size of the solutions and the non–inertness of the local solutions, we choose a different encoding, building a refined version of our CHAMs. Instead of using the $\overline{v}(\cdot,\cdot)/\underline{v}(\cdot,\cdot)$ pairs for the encoding of observed edges between nodes, we now encode these edges directly using symbols $e(v_i, v_j; v_k)$ for edges between the nodes v_i and v_j on a path to v_k. Hence, we get a different set of molecules:

Definition 5 *Let $V = \{v_1, v_2, \ldots, v_n\}$ be the set of nodes in the network to be examined. The molecules of the CHAMs $\mathcal{C}$ and $\mathcal{D}$ are taken from the set*

$$\begin{aligned} M \quad = \quad & \{\overline{v}(v_i, v_j)w \mid v_i, v_j \in V; w \in M'\} \\ \cup \quad & \{w\underline{v}(v_i, v_j) \mid v_i, v_j \in V; w \in M'\} \\ \cup \quad & \{\overline{v}(v_i, v_j)w\underline{v}(v_k, v_l) \mid v_i, v_j, v_k, v_l \in V; w \in M'\} \\ \cup \quad & \{e(v_i, v_j; v_k) \mid v_i, v_j, v_k \in V\} \\ \cup \quad & M', \end{aligned}$$

with the set of molecules M' being defined as

$$M' = \{v(v_i, v_j) \mid v_i, v_j \in V\}^*.$$

Hence, we just extend the set of molecules to also contain the encodings of edges. For an observed path

$$P = v_1 \to v_2 \to \ldots \to v_n$$

we now encode the solution

$$enc'(P) = \{\!|e(v_1, v_2; v_n), \ldots, e(v_{n-1}, v_n; v_n)|\!\}.$$

This way, we are able to formulate a bijective mapping between the encoding functions enc and enc'. Again, for a local observation $O = \{P_1, P_2, \ldots, P_m\}$, we encode the local solution as $enc'(O) = \uplus_{P \in O} enc'(P)$.

In the formulation of the reaction rule, we now allow non–inert molecules to *only* react with the basic edge molecules, while disallowing the edge molecules to react with other edge molecules.

Definition 6 *Let the set of molecules M be defined as above. The special reaction rule for CHAM $\mathcal{C}$ has the form*

$$u\underline{v}(v_i, v_k), e(v_i, v_j; v_k) \mapsto u\underline{v}(v_i, v_k), uv(v_i, v_k)\underline{v}(v_j, v_k), e(v_i, v_j; v_k).$$

Again, we do not "consume" the educts of the reaction to model a nearly infinite supply. The notion of global solution can be adapted from the corresponding definition in Section 3.1 in the same way as the definition of the local solution, i. e. by defining

$$enc'(N) = enc'(O_{v_1}) \uplus enc'(O_{v_2}) \uplus \ldots \uplus enc'(O_{v_n})$$

for some network N and observations $O_{v_1}, O_{v_2}, \ldots, O_{v_n}$.

Example 4 *Assume that the set of nodes V and the observation O_a are the same as in Example 1. The local solution for node a then is*

$$\begin{array}{rcl} enc'(O_a) & = & \{\!|e(a,b;b)|\!\} \\ & \uplus & \{\!|e(a,c;c)|\!\} \\ & \uplus & \{\!|e(a,b;d), e(b,d;d)|\!\} \\ & = & \{\!|e(a,b;b), e(a,c;c), e(a,b;d), e(b,d;d)|\!\}. \end{array}$$

With these definitions, we are able to perform the search for routes between nodes with CHAM $\mathcal{C}$ as well. The source and target for the search are supplied using the same method that was used in Section 3.1.

Example 5 *Let the encoding of the network be adapted from the one given in Example 2. To compute some of the routes between the nodes b and d, we follow the reaction*

$$\begin{array}{rl} & enc'(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ \mapsto & enc'(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)\underline{v}(d,d), v(b,d)\underline{v}(c,d)|\!\} \\ \mapsto & enc'(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)\underline{v}(d,d), v(b,d)\underline{v}(c,d)|\!\} \\ & \uplus \{\!|v(b,d)v(d,d), v(b,d)v(c,d)\underline{v}(d,d)|\!\} \quad , \\ \mapsto & enc'(N) \uplus \{\!|\underline{v}(b,d), \overline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)\underline{v}(d,d), v(b,d)\underline{v}(c,d)|\!\} \\ & \uplus \{\!|v(b,d)v(d,d), v(b,d)v(c,d)\underline{v}(d,d)|\!\} \\ & \uplus \{\!|v(b,d)v(c,d)v(d,d)|\!\} \end{array}$$

using in each step the special reaction rule from Definition 6. The result of the reactions is a solution containing the two inert molecules $v(b,d)v(d,d)$ and $v(b,d)v(c,d)v(d,d)$, which correspond to the paths $b \to d$ and $b \to c \to d$, respectively.

As we can see, the reaction in Example 5 is mostly equivalent to the one given in Example 3, except for the encoding of the network. We are able to state the connection between the two CHAMs $\mathcal{A}$ and $\mathcal{C}$ in the following theorems.

Theorem 4 *Let N be a network, containing nodes v_s and v_t. For each reaction*

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

performed by CHAM $\mathcal{C}$ we are able to find a reaction

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S'$$

with

$$S = \uplus_{m \in S'} \{|\phi(m)|\}, \qquad \phi(m) = \begin{cases} \overline{v}(v_i, v_k)\underline{v}(v_j, v_k) & m = e(v_i, v_j; v_k) \\ m & otherwise \end{cases}$$

for CHAM $\mathcal{A}$.
Proof: For each reaction of CHAM $\mathcal{C}$, we map each intermediate solution S'' by applying the function ϕ elementwise, yielding a reaction step inside CHAM $\mathcal{A}$. Doing this for the full reaction results in a reaction in left–to–right form for CHAM $\mathcal{A}$. □

For the reverse direction from CHAM $\mathcal{A}$ to $\mathcal{C}$, we focus on the "interesting" reactions in CHAM $\mathcal{A}$, which build to incomplete or complete paths.

Theorem 5 *Let N be a network with set of nodes V, containing nodes v_s and v_t. For each reaction*

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

performed by CHAM $\mathcal{A}$ containing molecules from the set

$$\begin{aligned} M'' \quad = \quad & \{v(v_s, v_t)w\underline{v}(v_j, v_t) \mid w \in \{v(v_i, v_t) \mid v_i \in V\}^*, v_j \in V\} \\ \cup \quad & \{v(v_s, v_t)wv(v_t, v_t) \mid w \in \{v(v_i, v_t) \mid v_i \in V\}^*\}, \end{aligned}$$

we are able to give a reaction

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S',$$

in CHAM $\mathcal{C}$ where S' contains the molecules from M' in the same multiplicities as S.

Proof: For each molecule from M' occurring in S, a chain of reactions in left–to–right form can be given. In each of the reaction steps, substitute $e(v_i, v_j; v_t)$ for each occurrence of $\overline{v}(v_i, v_t)\overline{v}(v_j, v_t)$ in the products and educts. Combining the resulting reaction chains provides a reaction in CHAM $\mathcal{C}$ leading to the required solution S'. □

Again, we have the problem that the process of reaction will not terminate because of an infinite supply of non–inert molecules. Similar to Section 3.1, we tackle this problem by constructing a fourth CHAM $\mathcal{D}$ based on CHAM $\mathcal{C}$, changing the special reaction rule.

Definition 7 *Let the set of molecules M be defined as in Definition 5. The special reaction rule for the third CHAM has the form*

$$u\underline{v}(v_i, v_k), e(v_i, v_j; v_k) \mapsto uv(v_i, v_k)\underline{v}(v_j, v_k), e(v_i, v_j; v_k).$$

We are able to state a connection between the reactions inside CHAM $\mathcal{C}$ and $\mathcal{D}$, too. As in Section 3.1, we first state the existence of normal form reactions.

Theorem 6 *Let N be a network, containing nodes v_s and v_t. Let S be a solution found in a reaction*

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

in CHAM $\mathcal{C}$. Then there exists a unique reaction in left–to–right form leading to the same solution S.
Proof: Straightforward. □

Now, we are able to state the connection between the two CHAMs in the following two theorems. The first one again states that each solution found by CHAM $\mathcal{D}$ can also be found by CHAM $\mathcal{C}$.

Theorem 7 *Let N be a network, containing nodes v_s and v_t. Let S be a solution found in a reaction*

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

containing a molecule m of the form

$$m = v(v_1, v_n)v(v_2, v_n) \ldots v(v_{n-1}, v_n)v(v_n, v_n),$$

in the CHAM $\mathcal{D}$. Then there exists a unique reaction in CHAM $\mathcal{C}$ leading to a solution S' containing m .
Proof: In each step of the reaction

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S,$$

add the educt $u\underline{v}(v_i, v_k)$ in the reaction step to the products of the step. The resulting chain of reactions will be a reaction inside CHAM $\mathcal{C}$. □

The second theorem parallels Theorem 3 in stating that each inert molecule found by CHAM $\mathcal{C}$ can also be found by CHAM $\mathcal{D}$.

Theorem 8 *Let N be a network, containing nodes v_s and v_t. For each reaction*

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

performed CHAM $\mathcal{C}$ containing an inert molecule m of the form

$$m = v(v_1, v_n)v(v_2, v_n) \ldots v(v_{n-1}, v_n)v(v_n, v_n),$$

we are able to find a reaction

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S'$$

in CHAM $\mathcal{D}$ with m being an element of S'.
Proof: As in the proof for Theorem 3, filter out of the chain of reactions

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

all those reactions not leading to educts used in the production of m. From all remaining reactions, remove the educts of the form $u\underline{v}(v_i, v_j)$ from the products of the reaction step. The resulting chain of reactions is a reaction inside CHAM $\mathcal{D}$. □

To close the square, we are also able to state a connection between the CHAMs $\mathcal{B}$ and $\mathcal{D}$. This is done in the next two theorems.

Theorem 9 *Let N be a network, containing nodes v_s and v_t. For each reaction*

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

performed by CHAM $\mathcal{D}$ we are able to find a reaction

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S'$$

with

$$S = \uplus_{m \in S'} \{|\phi(m)|\}, \qquad \phi(m) = \begin{cases} \overline{v}(v_i, v_k)\underline{v}(v_j, v_k) & m = e(v_i, v_j; v_k) \\ m & otherwise \end{cases}$$

for CHAM $\mathcal{B}$.
Proof: Along the lines given for the proof of Theorem 4. □

Theorem 10 *Let N be a network with set of nodes V, containing nodes v_s and v_t. For each reaction*

$$enc(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S$$

performed by CHAM $\mathcal{B}$ containing molecules from the set

$$\begin{aligned} M'' \quad = \quad & \{v(v_s, v_t)w\underline{v}(v_j, v_t) \mid w \in \{v(v_i, v_t) \mid v_i \in V\}^*, v_j \in V\} \\ \cup \quad & \{v(v_s, v_t)wv(v_t, v_t) \mid w \in \{v(v_i, v_t) \mid v_i \in V\}^*\}, \end{aligned}$$

we are able to give a reaction

$$enc'(N) \uplus \{|\underline{v}(v_s, v_t), \overline{v}(v_t, v_t)|\} \mapsto^* S',$$

in CHAM $\mathcal{D}$ where S' contains the molecules from M' in the same multiplicities as S.
Proof: Along the lines given for the proof of Theorem 5. □

While CHAM $\mathcal{A}$ is the one closest to our intuitive idea of finding routes by using DNA computing, CHAM $\mathcal{D}$ is the most accessible for practical implementations.

4 A TCP/IP based experiment

In order to show the applicability of our approach, a TCP/IP based experiment has been performed. For this experiment, a UNIX script was executed on 12 workstations distributed over Germany (8), Switzerland (2), the Netherlands (1) and Canada (1). The workstation in Canada

was hidden behind a firewall. It was able to find all other workstations, but none of the others was able to find the canadian computer.

The script made extensive use of the UNIX `ping` command. `Ping` is normally used to check the availability of a computer in a TCP/IP network. If used without options, `ping` indicates if a computer is alive or not. By using `ping` with the options `-s -Rv`, detailed information about the route used to check the availability of the computer can be retrieved. When executing our script, each workstation tried to ping all other workstations and retrieved detailed information about the trials. A section of the script reads:

```
echo ‘‘Station asterix.unibe.ch:’’ >> /tmp/results
ping -s -rv asterix.unibe.ch 56 10 >> /tmp/results
```

If the script is executed on the workstation `atlas.informatik.mu-luebeck.de`, the `ping` command aboove tries to reach `asterix.iam.unibe.ch` at the University of Berne in Switzerland. The results of the trial looks like this:

```
Station asterix.unibe.ch:
64 bytes from asterix.unibe.ch (130.92.64.4):
   icmp\_seq=0. time=1213.˜ms
IP options: <record route> 141.83.100.3, 141.83.100.1,
   cisco-MU-Luebeck.medinf.mu-luebeck.de (188.1.132.213),
   XR-Hamburg1.WiN-IP.DFN.DE (188.1.3.250),
   ZR-Hamburg1.WiN-IP.DFN.DE (188.1.144.21),
   ZR-Hannover1.WiN-IP.DFN.DE (188.1.144.25),
   ZR-Koeln1.WiN-IP.DFN.DE (188.1.144.53),
----asterix.unibe.ch PING Statistics----
10 packets transmitted, 1 packets received,
90% packet loss round-trip
(ms) min/avg/max = 1213/1213/1213
```

In this special case, the route only has been recorded up to the node `ZR-Koeln1.WiN-IP.DFN.DE`, and is lost from this point on.

The results of the trials have been analysed, and 179 additional computers used as gateways for the ping information have been detected. So the virtual network to be analysed consisted of 11 active nodes which could make experiments and which could be reached by a `ping` command, 1 active node which could not be reached and 179 passive gateway nodes, making a total of 191 nodes.

For this network, we constructed the CHAM and started to analyse the model. We found several new paths between the active nodes which should be possible in the real network but which were not observed during our trials. Due to complexity and space limitation the constructed CHAM cannot be presented here. However, the local view of `atlas.informatik.mu-luebeck.de`is as

follows:

$$
\begin{aligned}
\{\!| \quad & e('143.83.100.3','141.83.100.1';'130.92.64.4'),\\
& e('143.83.100.1','188.1.132.213';'130.92.64.4'),\\
& e('188.1.132.213','188.1.3.250';'130.92.64.4'),\\
& e('188.1.3.250','188.1.144.21';'130.92.64.4'),\\
& e('188.1.144.21','188.1.144.25';'130.92.64.4'),\\
& e('188.1.144.25','188.1.144.53';'130.92.64.4'),\\
& e('188.1.144.53','\mathit{uncharted\ subnet}';'130.92.64.4')\\
& e('\mathit{uncharted\ subnet}','130.92.64.4';'130.92.64.4')\\
|\!\}
\end{aligned}
$$

It is obvious that the local view is just a rewriting of the ping result, except for the "uncharted subnet" introduced after the gateway `ZR-Koeln1.WiN-IP.DFN.DE`.

However, at the time of writing this paper the analysis of the results of this experiment was not fully completed. We will give more detailed information about the number of additional paths found by executing the CHAM.

5 Conclusion and Outlook

We presented a solution for the problem of discovering the behaviour of a network from local observations. Our work is based on the chemical computation metaphor presented in [6] and the idea of molecular computation presented in [1]. Four different CHAMs have been defined as solutions to our problem and their advantages and drawbacks have been discussed. The applicability of our approach has been shown by an experiment in which we have constructed a CHAM from local observations of UNIX workstations for a virtual TCP/IP network with 191 nodes. As a first result of this experiment the CHAM found additional routes through the network which were not observed during the experiment.

Our future work is directed towards several goals: We intend to provide software support for the analysis and visualisation of CHAM based network models. This work will be based on some prototype tools which we have already developed for the analysis of the described experiment. Our theoretical work will concentrate on applying our network model to more sophisticated problems. A starting point will be to study whether our approach is suitable to predict a network behaviour if nodes are added to or deleted from a network. Also, higher concepts in CHAMs as, for instance, the airlock operator from [6] have to be examined. With respect to this operator, it is planned to look at a relation between the CHAMs we presented and models for process algebras as given in [6, 7] to generate simulation models from local observations.

Acknowledgements

The presented work has been supported by Siemens Switzerland. The authors would like to thank Dieter Hogrefe (ITM Lübeck) for supporting this work, Iwan Nussbaumer and Charles Zehnder (Siemens Switzerland) for many valuable discussions, Beat Koch and Michael Schmitt (ITM Lübeck) for careful proofreading, and all colleagues for helping to perform the experiment by running our script on their workstations.

References

[1] Adleman, L.: Molecular Computation of Solutions to Combinatorial Problems. Science 266 (November 1994), pp. 1021–1024.

[2] Banâtre, J.–P., Coutant, A., Le Métayer, D.: A parallel machine for multiset transformations and its programming style, Future Generation Comput. Systems 4 (1988), pp. 133–144.

[3] Banâtre, J.–P., Le Métayer, D.: The Gamma model and its discipline of programming. Science Comput. Programming 15 (1990), pp. 55–77.

[4] Beaver, D.: Computing with DNA. Journal of Computational Biology (2:1), Spring 1995.

[5] Berry, G., Boudol, G.: The Chemical Abstract Machine, POPL 90 (1990), pp. 81–94.

[6] Berry, G., Boudol, G.: The Chemical Abstract Machine, Theoretical Computer Science vol. 94, May 1992.

[7] Boudol, G.: Some Chemical Abstract Machines. In *A Decade of Concurrency*, number 803 in Lecture Notes in Computer Science, pp. 92–123. Springer-Verlag, May 1994.