Constan growth in demand for computational power requires advances in the internal mechanisms of multiprocessor computing structures. Such architectures may include many (sometimes, even millions of) processors performing processing tasks. Each technique that increases efficiency leads to significant benefits in operational energy and task execution time. Due the scale of multiprocessor computing structures, the importance of achieving faster and efficient systems is invaluable. In this paper, we present two different approaches for processing tasks on multiprocessor architectures: Hardware-Physical (H-Phy) and Overlay-Network-on-Chip (Overlay-NoC). Both methods are described and compared. We also present the research plan, models, simulation assumptions, and results of research. The paper is summarized with conclusions and future work plan.

Keywords—CMP, Overlay-NoC, Simulation System.

I. INTRODUCTION

High performance computing architectures are currently under extensive research with an aim of achieving exascale systems (capable of an exaflop, or $10^{18}$ floating point operations per second) by 2020. Simply attempting to scale architectures to billions of processors is not sufficient. Increasing the efficiency of individual processing nodes has to be addressed [1]. Individual processing nodes currently used and developed are Chip Multiprocessors (CMPs) with number of cores ranging between few and tens [2]. The cores within the CMPs are connected through Network-on-Chip (NoC). For NoC implementation, low dimensional topologies are used, especially 2D-mesh and 2D-torus [3]. Physical structure of modern CMPs with NoC is a subject of intensive research, e.g. [4], [5] and [6]. Also, computation problems based on distributed structures on higher architecture levels is also the subject of wide research today [7].

Except for well-designed internal CMP architecture, performance of the CMP depends on efficient utilization of the cores. Most processor allocation algorithms concentrate on allocation that targets effective processor utilization while minimizing execution time. The authors of [8] created a model and proposed algorithms for minimizing the energy usage of interconnected processors. They define the energy costs for processor, assuming that they are fully interconnected. It is similar to our model. However, we consider energy consumed by NoC as well [3], [4], [9] and [10]. Processor cluster distributed structure was presented in [11], where authors have proposed 12-processor clusters connected into distributed systems. However, in this system clusters are connected among each other using regular TCP/IP network connections. Connections inside the clusters are implemented using regular buses, that increase transport efficiency, but transportation energy is not considered. Another approach for energy management was presented in [12], where the distributed system contains many FPGA boards. Processors sharing the same board may communicate among each other using shared memory implemented on the chip as well. Ethernet-based network with TCP/IP stack is used for communication between many boards. The system uses intra- and inter-chip data exchange. The energy management, unlike in previous approaches based on scheduling control, was achieved by controlling the processors’ core frequency scaling. Each processor contains temperature sensor and can be controlled separately (in terms of frequency domain). We consider our model as extendable with such techniques in future work, where additional minimization of energy consumption can be achieved.

Simulation is a common technique used in research for all kinds of systems, including distributed systems. Some example implementations of this kind of systems are shown in [9], [13], [14], [15] or [16]. Authors researched various techniques with regard to allocation-related performance, which could also be interesting subject of research with regard to energy consumption. The simulation system presented in this paper shares some ideas with the approaches presented in [9] and [13]: Modular design, data generator, and framework construction providing interface to implement various operating algorithms. The idea of time slicing is used in [14], that brings the experiments even more close to simulated structures.

In this paper, a system capable of analyzing a CMP using two approaches is presented: 1) Hardware-Physical (H-Phy) approach and 2) a new, Overlay-NoC approach. Both of them use the same input data set for processing, but additionally they apply approach-specific factors. Thus approaches differ not only by algorithms used. H-Phy and Overlay-NoC approaches were developed independently and
for final versions, simulations were conducted. Based on the presented evaluation system, examples of experiments are conducted and results are presented.

The reminder of this paper is organized as follows: Section II presents a simulated CMP and H-Phy approach, while the overlay tactic is described in Section III. Section IV contains a description of experiments and analysis of results. Conclusions and final remarks are presented in Section V.

II. CMP AND H-PHY APPROACH

A physical layout of the simulated CMP is presented in Fig. 1. A detailed description can be found in [4]. The chip area is divided into tiles, that ensures scalability and effective use of resources available on the chip. Each tile contains networking elements -- router, networking interface, network channels -- and Processing Elements (PEs) -- processor, cache memory, etc. Tiles are connected among each other by a network Router. We consider a homogenous architecture, where all PEs in CMP are the same (they have the same computational power). The efficient use of PEs is done by utilizing a Processor Allocator (PA) and Job Scheduler (JS). Both PA and JS and implemented in hardware and placed as tile on the same die as the PEs in the CMP, that delivers better performance and efficiency [4].

A. Simulation Methodology

The simulation starts when JS receives request to allocate a job. The job is described by the size of the requested subgrid. In the H-Phy approach, we consider contiguous processor allocation strategy, where the processors allocated to a job are physically adjacent and have the same topology as NoC. JS is responsible for job scheduling that deals with the selection of the job to be executed next. In our system, the JS processes jobs in First Come First Served fashion. The scheduled job is moved to a PA, which assigns the job to available PEs according to allocation algorithm. We use two best allocation schemes: IFF algorithm for mesh [5] and BMAT technique for torus [3]. Once the PA finds available PEs to accommodate the job, the PA sends an allocation message to PEs to reserve them for the job. The jobs are allocated in such a manner that they do not overlap with each other and if they are allocated, they run until completion. If there is no free PEs, the PA waits until another job will release some PEs. After execution, PEs send a release message to the PA, which updates the status of processors. All messages in the system are sent by the implemented NoC. We assumed that allocation and release messages take one flit, e.g. if job requires 4 processors, 4 flits have to be sent from a PA to all 4 PEs assigned to the job.

B. Assumptions and Description

The model for H-Phy approach is defined as follows:

Indices:

| v, w | 1, 2, ..., V | PEs |
| b   | 1, 2, ..., B | job to JS |
| s   | 1, 2, ..., S | sizes of jobs |
| t   | 1, 2, ..., T | time slots |

Binary variables:

\[ q_{bs} = 1 \text{ when job } b \text{ has horizontal size } s \text{ or less, } 0 \text{ otherwise} \] (binary)
\[ r_{bs} = 1 \text{ when job } b \text{ has vertical size } s \text{ or less, } 0 \text{ otherwise} \] (binary)
\[ y_{bMvt} = 1 \text{ when job } b \text{ is sent from PA to PE } v \text{ in time slot } t, 0 \text{ otherwise} \] (binary)
\[ x_{bvt} = 1 \text{ when job } b \text{ is computed at PE } v \text{ in time slot } t \]
\[ i, j = 1, 2, ..., N \] indices for position of PE
\[ g_{vij} = 1 \text{ when PE } v \text{ resides at position } i,j \text{ (i=horizontal, j=vertical) in mesh or torus structure} \]

Constants

- \( E_{bi} \): energy consumption to send one bit from \( v \) to \( w \)
- \( W \): word length
- \( X, Y \): size of mesh/torus (horizontal/vertical)

Criterion function

minimize \( F = 2W \sum_{v,w} \sum_{b} y_{bMvt} E_{bi} M_{v} \sum_{s} q_{bs} \sum_{r_{bs}} r_{bs} \)

Constraints

all jobs have to be computed:
\[ \sum_{v,w} x_{bvw} = B \] (1) each job is computed once:
\[ \sum_{v,w} x_{bvw} = 1 \] \( b = 1, 2, ..., B \) (2) PEs do not exchange data packets between each other:
\[ \sum_{v} \sum_{w} \sum_{e} y_{bMve} = 0 \] \( v \neq w \neq M \) (3) Job is allocated to PE which is not occupied:
\[ \sum_{v} x_{bvw} = 1 \] \( t = 1, 2, ..., T, \quad v = 1, 2, ..., V \) (4)

Mesh specific constraints:

Job is allocated to adjacent PEs, job must not overlap mesh:
\[ \sum_{v} \sum_{w} \sum_{e} g_{(v+e)(w+e)} = \sum_{v} q_{bs} \sum_{r_{bs}} \]
\[ 1 \leq t \leq T, \quad b = 1, 2, ..., B, \quad 0 \leq e \leq \sum_{v} q_{bs} \quad 0 \leq f < \sum_{r_{bs}} \]
A job \( b \) may contain one or many tasks that are adjacent to each other. A job has a shape that is a subgrid of the NoC topology, and it is described by the size of the subgrid it requires (Fig. 2). All PEs in CMP are the same and each PE may process only one task in the same time (4) so, for jobs containing more than one task, more PEs are needed, e.g. for the job from Fig. 2, six adjacent PEs are needed and they assume the shape as illustrated in the Fig. Once job is allocated to PEs it runs until completion. PEs are connected among each other by NoC, the one researched in this paper constitutes the communication network, as it is described in Section II and shown in Fig. 1. In the overlay approach a non-contiguous allocation strategy is used – job can be executed on multiple disjoint smaller subgrids. It allows dividing a job rather than waiting until a single subgrid of the requested size is available. Thus, we assume that input jobs are divisible. The considered NoC topology is 2D-Mesh and 2D-Torus structure. In this approach, a heterogeneous CMP architecture can be considered (PEs do not have to be the same), i.e. the cost data unit computation may be different among them. This way we can define the structure containing PEs with different energy consumption and different computing power (though we are able to set costs and power for equal values for all PEs to get the uniform parameters for all of them). The presented mathematical model is valid for the heterogeneous architecture. However, to be consistent with the H-Phy approach presented in Section II, the simulations are conducted for the homogenous structure, where all PEs are the same.

### Torus specific constraints:

Job is allocated to adjacent PEs, job can overlap torus:

\[
\sum \sum N_{\text{flit}} \sum_{i=1}^{b} g(i+e \mod \text{tile}, (j+f \mod \text{tile})m) = \sum q_{\text{bus}} \sum r_{\text{bus}}
\]

\(1 \leq i < X \sum q_{\text{bus}}, 1 \leq j < Y \sum r_{\text{bus}}\) \hspace{1cm} (5)


\[
\text{Job } b \quad (\text{where } \% \text{ symbol denotes modulo division})
\]

\[
\text{Fig. 2. A job } b \text{ that contains 6 tasks.}
\]

### C. NoC and Energy Model Used

We investigate NoC architectures with:

1. 2D-Mesh and 2D-Torus topologies,
2. Virtual-channel flow control,
3. Dimensional Order Routing (DOR)[4].

Each NoC node consists of a Virtual Channel (VC) router (R in Fig. 1). A packet traversing from a node (tile) \( v \) to neighboring tile \( w \) (it is one NoC channel – 1 hop) needs to be processed by the VC router where next destination node is selected, and it needs to traverse NoC Channel. The average energy consumption in \( \text{flit} \) of sending one bit of data from tile \( v \) to tile \( w \) is expressed by:

- For 2D-Mesh:
  \[
  E_{\text{bus}}^{v,w} = 0.98(N_{\text{hop}}^{\text{VC}} + 1) + 0.57N_{\text{hop}}
  \]
  \( \hspace{1cm} (7) \)

- For 2D-Torus:
  \[
  E_{\text{bus}}^{v,w} = 0.98(N_{\text{hop}}^{\text{VC}} + 1) + 0.75N_{\text{hop}}
  \]
  \( \hspace{1cm} (8) \)

where \( N_{\text{hop}}^{\text{VC}} \) is the number of VCs traversed by a packet between tile \( v \) and \( w \), and \( N_{\text{hop}} \) is the number of physical channels traversed by the packet. The values 0.98 and 0.57 are obtained based on hardware implementation of NoC on an FPGA device [18].

We consider the system built with Intel Core i5-660 processors having clock 3.6 GHz. These units include two physical cores inside, but we treat one i5 chip as one PE. According to Intel technical specifications [17] i5-660 have Thermal Design Power (TDP) equals to 73W. Computing power expressed in GFLOPs equals to 29. We use TDP as the operating power of cluster processors to give good estimate of energy consumption. We convert the TDP into energy consumed in a cycle \( E_{\text{c}} \) [18] according to formula:

\[
E_{\text{c}} = \frac{TDP}{F_{\text{max}}}[\mu J]
\]

where \( F_{\text{max}} \) is the maximum frequency of Intel Core i5-660 processor in [MHz].

### III. OVERLAY-NOC BASED APPROACH

On-chip processors (cores) are connected by NoC that constitutes the communication network, as it is described in Section II and shown in Fig. 1. In the overlay approach a non-contiguous allocation strategy is used – job can be executed on multiple disjoint smaller subgrids. It allows dividing a job rather than waiting until a single subgrid of the requested size is available. Thus, we assume that input jobs are divisible. The considered NoC topology is 2D-Mesh and 2D-Torus structure. In this approach, a heterogeneous CMP architecture can be considered (PEs do not have to be the same), i.e. the cost data unit computation may be different among them. This way we can define the structure containing PEs with different energy consumption and different computing power (though we are able to set costs and power for equal values for all PEs to get the uniform parameters for all of them). The presented mathematical model is valid for the heterogeneous architecture. However, to be consistent with the H-Phy approach presented in Section II, the simulations are conducted for the homogenous structure, where all PEs are the same.

#### A. Simulation Methodology

The overlay approach includes three ways to achieve the result: i) optimal solutions ii) heuristic static-based solutions iii) heuristic simulation solutions. In this paper, we use the third approach, that gives us results reflecting H-Phy structure modeled during the research and described in Section II. Simulator uses the concept of time scale slicing and the message exchanging architecture. This makes it scalable and extendable for new mechanisms. The system also includes robust input data management, the trend-research module and scenario driven experiment capabilities. The optimal solution approach could not be used even as the comparing pattern, due to large scale of input data sets researched.

#### B. Assumptions and Description

For each processor pair the distance is known (which is interpreted as number of hops between two processors). The distance also refers to NoC energy cost, as for longer distances more energy has to be consumed to transfer data, so we define the following cost parameter for each PE pair:

\( k_{v,w} = \text{const}, \) which is the energy cost of sending the data.
packet between tiles \( v \) and \( w \). The parameters defined for each PE \( v \) are: computing power \( p_v = const \) and computation cost (for one data unit) \( c_v = const \). The overlay NoC and connected PEs also have limited data transfer capabilities. Each PE \( v \) is able to send maximum \( u_v = const \) data packets and receive maximum \( d_v = const \) data packets in a given period of time \( t \). The time scale is divided into \( T \) time slots, what is the additional constraint for the problem solved – it must be finished before limited \( T \) amount of time. The PE structure performs the computation and results’ distribution tasks, so the overall structure’s energy cost consists of computation cost and data transfer cost. Finally, our complete model for Overlay-NoC based processor structure is as follows:

### Indices
- \( b = 1, 2, \ldots , B \) indices for data packets blocks (problem contains \( B \) 1x1 tasks)
- \( t = 1, 2, \ldots , T \) indices for time slots (problem has to be resolved in maximum \( T \) time slots)
- \( v, w = 1, 2, \ldots , V \) indices for PEs
- \( M \) special unit: Processor Allocator (PA)

### Constants
- \( c_v \) cost of task computation at PE \( v \) (energy)
- \( k_{wv} \) cost of packet transfer between tiles \( w \) and \( v \) (energy)
- \( p_v \) computation power of PE \( v \)
- \( d_v \) packet receive limit of PE \( v \)
- \( u_v \) packet send limit of PE \( v \)

### Binary variables
- \( x_{bv} = 1 \) when task \( b \) is computed at PE \( v \); 0 otherwise (binary)
- \( y_{bv} = 1 \) when task \( b \) is transferred from PE \( v \) to PE \( w \) in time slot \( t \); 0 otherwise (binary)

### Criterion function
The criterion function (computation + transfer) for the model is formulated as:

\[
\min_{x,y} F = \sum_{b,v} x_{bv} c_v + \sum_{v,w} \sum_{t} y_{bvt} k_{wv}
\]  

(10)

### Constraints
We have to define the following constraints, to determine the additional assumptions to make the model complete:
- Each PE has to compute at least one task:
  \[
  \sum_{b} x_{bv} \geq 1 \quad v = 1, 2, \ldots , V
  \]  
  (11)
- Each task may be computed only on one PE:
  \[
  \sum_{v} x_{bv} = 1 \quad b = 1, 2, \ldots , B
  \]  
  (12)
- Each PE has limited computation power:
  \[
  \sum_{b} x_{bv} \leq p_v \quad v = 1, 2, \ldots , V
  \]  
  (13)
- PE’s sending data link capacity:
  \[
  \sum_{w} \sum_{t} y_{bvt} \leq u_v \quad w = 1, 2, \ldots , V \quad t = 1, 2, \ldots , T
  \]  
  (14)
- PE’s receiving data link capacity:
  \[
  \sum_{w} \sum_{t} y_{bvt} \leq d_v \quad v = 1, 2, \ldots , V \quad t = 1, 2, \ldots , T
  \]  
  (15)
- Data packets containing tasks are sent from PA to PEs, each such packet is sent once:
  \[
  \sum_{b} \sum_{v} y_{bvt} = B
  \]  
  (16)

The result of task computation has to be sent back enclosed in packet \( b \) from computing PE \( v \) to PA:

\[
\sum_{v} \sum_{t} y_{bvt} x_{bv} = B
\]

(17)

PEs do not exchange data packets between each other:

\[
\sum_{v} \sum_{w} \sum_{t} y_{bvt} = 0 \quad v \neq w \neq M
\]

(18)

To keep the model clear, we simplify and disregard the release message sent from each PE to PA. It is included in \( c_v \), because it is constant for each PE (the network distance from PA to each PE does not change). Unlike the H-Phy model, mesh and torus specific elements are provided by constant values of input data. Presented model is applied as rule set to the simulation system.

### C. NoC and Energy Model Used
As an NoC and energy model in the overlay approach, we assume the model presented in Section II C.

### IV. EXPERIMENTATION SYSTEM AND RESULTS
The evaluation of the considered simulation approaches was done in an experimentation system with logical structure presented in Fig. 3.

![Fig. 3. Block-diagram of the experimentation system as input-output system.](image-url)

The elements of the system are:
- **Input – \( A \)**: Queue with jobs.
- **Problem parameters – \( P \)**: Approach used (H-Phy or Overlay-NoC); \( P \): Size of the evaluated CMP; \( P \): Topology (2D-Mesh or 2D-Torus).
- **Outputs – \( O \)**: Computation Energy; \( O \): NoC Energy.

Experimental research performed for both approaches included the experiments both for mesh and torus CMPs. In the first experiment, we investigated 10x10 and 15x10 CMP structures. The randomly generated queues (discrete uniform distribution) contained 1,000 jobs that varied in size. Considering the number of single cores requested, the jobs in the queues used in experiments required 6,210 and 8,798 cores. Tasks for H-Phy approach were used in original size, the Overlay-NoC approach required to divide them into 1x1 tasks. To calculate computation energy, we assumed that one task for completion requires 4 cycles [19].

The average cost of transfer of one data packet between two processors in the whole structure equals:
- For 10x10 structure: 0.48 nJ (mesh) and 0.28 nJ (torus),
- For 15x10: 11.57 nJ (mesh) and 6.27 nJ (torus).
The experiment was performed for uniform energy costs, that models the cluster containing same processors. However, further research including various types of processors is planned.

As we can see in Table I, computation energy consumed by PEs is significantly higher in comparison to NoC energy. In H-Phy case, the computation energy is on average 98 times higher than NoC energy, while in overlay approach it is 110 times. The result is not a surprise and it proves the correctness of the models, since similar outcomes were reported in [18]. Both simulation tactics confirmed significant advantage of NoC based on torus topology (Table I and Fig. 4). In the H-Phy approach, average NoC energy consumption for meshes was 59% higher in comparison to torus topology. Similarly for the overlay tactic, the ratio was 63%. Additionally, average NoC energy consumption using the H-Phy approach was 11% higher than in the overlay case. It is correct since the overlay approach uses non-contiguous allocation strategy while contiguous scheme is used in the H-Phy method. This value reflects the profit achieved by the ability to divide tasks into single PE size. This aspect is investigated further by the following experiment. Three sets of input data are prepared: S1 containing 8,192 tasks of 1×1 size; S2 containing 2,048 2×2 tasks; S3 containing 512 4×4 tasks. Next, these sets were processed both using contiguous H-Phy approach and non-contiguous Overlay-NoC approach. Thus we observe the influence of task size for H-Phy mode compared with non-contiguous approach.

In the second experiment, we evaluated 10×10 CMP. We were considering jobs, where the total number of required tasks (cores) is 10,800. However, in H-Phy approach, the amount of jobs for allocation varied with the size of the jobs, so that we had series 10,800 1×1 jobs, 2,700 2×2 jobs, 1,200 3×3 jobs and 675 4×4 jobs. Similarly, as in the first experiment, to calculate computation energy we assumed that one task requires 4 cycles for completion [19]. E.g., one job 3×3 required 3×3×4 = 36 cycles. Results of the experiment are summarized in Table II, where the time of execution column shows the total number of cycles needed to find a free subgrid for a job, send the allocation message to reserved processors, execute the job and send release message to inform PA that the job has been executed. As we can see, in H-Phy approach, energy consumed by NoC depends on size of the jobs. Jobs 1×1 require the highest amount of NoC energy for 2D-Mesh structure; similarly, time of execution was higher. In 2D-Torus case, the size 1×1 became the most energy efficient; however time of execution remained weak as well. Generally, in H-Phy strategy, allocation efficiency depends on the size of the jobs and the CMP structure. Some additional dependencies in job allocation problems can be found in [9]. The Overlay-NoC approach yielded similar results for the second experiment as H-Phy, despite slightly different assumptions. This convinces us of the good quality of the proposed approaches and suggests that the assumptions in the problem significantly influences the result. The Overlay-NoC was performed only for 1×1 tasks, as bigger tasks would be divided into 1×1 anyway.

| TABLE I. ENERGY CONSUMPTION BASED ON THE APPROACH USED AND TOPOLOGY OF CMP. |
|---------------------------------|----------------|----------------|----------------|----------------|
|                                 | H-Phy | Overlay-NoC | Mesh | Torus | Mesh | Torus | Mesh | Torus |
| B=6210                          | 503.70 | 503.70 | 503.70 | 503.70 |
| CMP 10×10                        | 6.28  | 3.78  | 5.66   | 3.38   |
| Total Energy [µJ]               | 509.98 | 507.48 | 509.36 | 507.08 |
| B=8798                          | 713.61 | 713.61 | 713.61 | 713.61 |
| CMP 10×10                        | 8.80  | 5.34  | 8.03   | 4.80   |
| Total Energy [µJ]               | 722.41 | 718.95 | 721.64 | 718.41 |
| B=8798                          | 713.61 | 713.61 | 713.61 | 713.61 |
| CMP 15×10                        | 10.80 | 7.37  | 9.90   | 6.31   |
| Total Energy [µJ]               | 724.41 | 720.98 | 723.51 | 719.92 |

![Fig. 4. NoC Energy comparison for conducted experiments.](image)

<table>
<thead>
<tr>
<th>TABLE II. ENERGY CONSUMPTION FOR OVERLAY-NoC AND H-PHY APPROACHES IN 10×10 CMP WITH 10,800 TASKS IN FUNCTION OF THE SIZE OF JOB.</th>
</tr>
</thead>
<tbody>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td>Overlay-NoC</td>
</tr>
<tr>
<td>Jobs 1×1</td>
</tr>
<tr>
<td>Torus</td>
</tr>
<tr>
<td>H-Phy</td>
</tr>
<tr>
<td>Jobs 1×1</td>
</tr>
<tr>
<td>Torus</td>
</tr>
<tr>
<td>H-Phy</td>
</tr>
<tr>
<td>Jobs 2×2</td>
</tr>
<tr>
<td>Torus</td>
</tr>
<tr>
<td>H-Phy</td>
</tr>
<tr>
<td>Jobs 3×3</td>
</tr>
<tr>
<td>Torus</td>
</tr>
<tr>
<td>H-Phy</td>
</tr>
<tr>
<td>Jobs 4×4</td>
</tr>
<tr>
<td>Torus</td>
</tr>
</tbody>
</table>
V. CONCLUSIONS

In this paper, we have proposed two completely different approaches to modeling and simulation of NoC-based CMPs: Already known an H-Phy approach, and a new Overlay-NoC. The H-Phy strategy models contiguous processor allocation algorithms, while the overlay scheme is designed for non-contiguous allocation. We have proposed a simulation system, where both approaches can be used and evaluated. We have also presented newly constructed Mixed Integer Programming optimization models for both H-Phy and Overlay-NoC approaches.

Experimental results presented in the paper confirm lower NoC energy consumption of CMP when torus topology is used. Similarly, the NoC energy consumption using non-contiguous processor allocation strategy is lower in comparison to contiguous approach. These observations were confirmed for both H-Phy and overlay strategies. The presented evaluation system together with results also reveal that the overlay NoC simulation approach models modern CMPs as accurately as a H-Phy tactic. It gives enormous flexibility and possibilities of modeling NoC-based CMPs and is a promising approach for further research.

REFERENCES

