Computing within Materials: Self-Adaptive Materials and Self-organizing Agents

Stefan Bosse, University of Bremen, Dept. 3, 28359 Bremen, Germany

Dirk Lehmhus, Fraunhofer IFAM, 28359 Bremen, Germany

Atra Gemilang, University of Bremen, Dept. 4, 28359 Bremen, Germany

Abstract: Materials Informatics addresses commonly the design of new materials using advanced algorithms and methods from computer science like Machine Learning and Data Mining. Ongoing miniaturization of computers down to the micro-scale-level enables the integration of computing in structures and materials that can be understand as Materials Informatics from another point of view. There are two major application classes: Smart Sensorial Materials and Smart Adaptive Materials. The latter class is considered in this work by combining self-organizing and adaptive Multi-agent Systems with materials posing changeable material properties like stiffness by actuators. It is assumed that the computational part of this micro-scale Cyber-Physical-System is entirely integrated in the material or structure as a distributed computer composed of a network of low-resource computers. Each node is connected to sensors and actuators. Actually only macroscopic systems can be realized. Therefore a multi-domain simulation combining computational and physical simulation is used to demonstrate the approach and to evaluate self-adaptive algorithms.

1. Introduction

Load-bearing structures are typically designed towards relevant load cases assuming static shape and fixed sets of materials properties decided upon during design and materials selection. New technologies enabling the design of structures that could change local properties in service in response to load change could raise additional weight saving potentials, thus supporting lightweight design and sustainability. Materials with such capabilities must necessarily be composite in the sense of a heterogeneous build-up, exhibiting, e.g., an architecture consisting of networks with numerous active cells providing sensing, signal and data processing, communication, and actuation/stimulation capability **[1]** forming Adaptronic Structures **[2]**. One example for such a material is a special class of polymers being capable to change their elasticity based on the influence of optical, thermo, or electrical fields. One concern regarding active smart cellular structures is the correlated and self-organizing control of cells' responses, and the underlying informational organization providing robustness and real-time capabilities.

We suggest a hybrid approach that combines mobile and reactive self-organizing Multi-agent Systems (MAS) **[3]** processed within the material and Machine Learning.. The goal of the MAS is basically solving a minimization problem, but with an incomplete (partially unknown) world model The minimization function relies on unreliable and noisy sensor input data. In it, the MAS' task is to analyze loading situations based on sensor data and negotiate matching spatial redistributions of material properties like elastic modulus to achieve higher-level optimization aims like a minimum of the total strain energy within the structure, or a reduction of peak stress levels. The associated Machine Learning approach (not considered in this work) would be employed to recognise loading situations for which optimized solutions exist and in such cases to adapt the MAS system to directly enforce the respective property distribution. Furthermore, reinforcement learning can improve the overall response of the system under

varying environmental conditions and to find an optimal minimization solution. In the present study, a proof of concept of the approach is presented which combines finite element methods (FEM), Multi-body physics (mass-spring model), and MAS simulation, with the former primarily taking the place of the physical structure.

FEM simulations are used to investigate suitable minimization approaches on algorithmic level and for off-line training of the MAS prior to its deployment in the real or simulated structure. We provide a multi-domain simulation framework. In contrast, a Multi-body physics model provides a simplified model for the MAS simulation and evaluation with real-time resolution composing the adaptive material of non-deformable mass nodes connected by adaptive springs (variable stiffness and damping parameters).

The integration of computation into materials as another view of Materials Informatics is an enabling technology among materials with controllable properties for the design of future smart materials and structures. Integration of computation introduces hard constraints: Low Size (<1mm^3); Low-Power (<1mW); Unreliability; and no maintenance inside. We developed agent processing platforms that are capable to be integrated in materials. The suitability of the ICT and the structure adaptability approaches are evaluated with a case study.

2. Smart Materials: Fusion of Sensorial- and Adaptive Materials

Any real world load-bearing structure may see different scenarios including misuse, which might be impossible to capture/define in the design stage. Such unforeseen loads may in the worst case lead to immediate failure - in other cases, they might just wear out a structure, causing it to fail prematurely and/or in places determined by unexpected local load histories. A smart adaptive material is capable of adjusting its mechanical characteristics - like stiffness - to external loads could actively manage this local load history. It could protect areas already worn out and distribute loads to others instead.

In our understanding, a smart material provides the following major features:

1. Perception using various kinds of sensors, e.g., measuring of strain, displacement, temperature, pressure, forces;
2. Changing of local material and structure properties by actuators, e.g., stiffness or damping variation;
3. Integrated Information and Communication technologies (ICT);
4. Distributed Approach: Local sensor processing and actuator control - Global cooperation and coordination.

The general model of a smart adaptive material is shown in Fig. **1**. It is assumed that the smart material consists of volume elements (bounded regions of the material) that are connected with neighbourhood elements via links. The links should provide some kind of sensing (e.g., measuring the strain or displacement along the link main axis) and some kind of material or structure control providing a controllable actuator (e.g., modifying the stiffness of the link). A link can be a discrete part or a continuous region of the material. A set of actuators is connected to each node. Two nodes share an actuator (e.g., a damped spring).

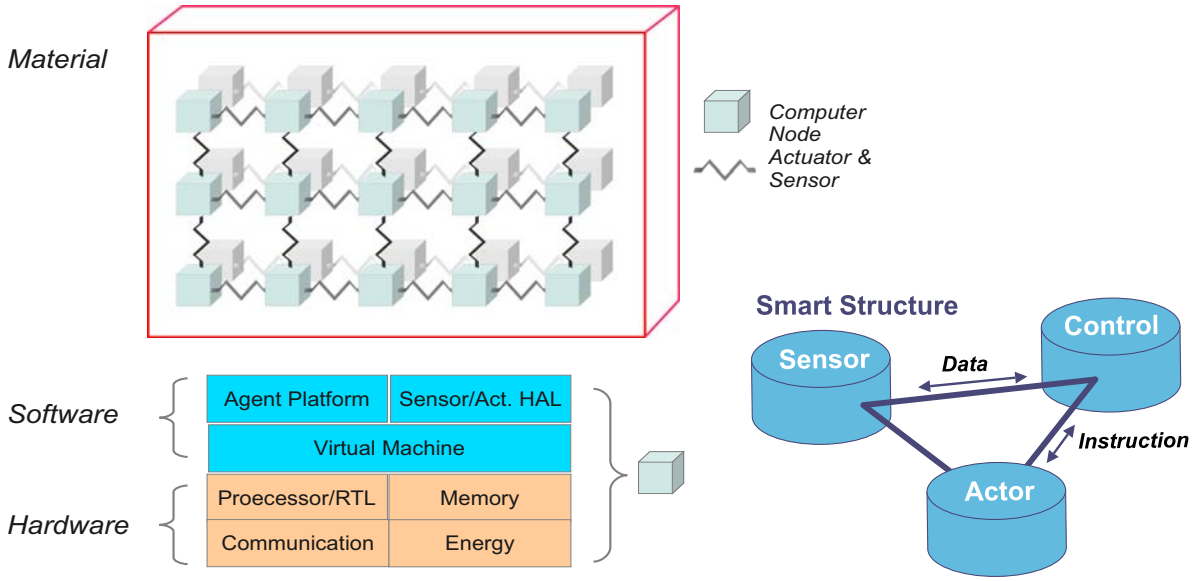


Figure 1. (Left) General architecture of a Sensorial- and Adaptive Material = Smart Material (Right) Functional Decomposition: Sensing, Acting, Processing ↔ Data + Instruction Streams

Each element contains an embedded computer that is considered as a node in a mesh-grid communication network. Each node provides some kind of data processing (processor or digital logic RTL), data and program memory, communication, energy supply and energy management. Since the distributed sensing and control of the material should be performed by an agent-based approach each node have to provide an agent processing platform (APP). Agents have to access sensors and actuators by an hardware abstraction layer (HAL), optimally provided by the APP.

It is assumed that the nodes are organized in an ad-hoc way. Technical failures of single nodes are considered as the normal case that may not effect the operation of the smart material satisfying some global objective function. That means a self-organizing and self-adaptive approach with respect to the connectivity structure has to be used to created some kind of a holonic system and is the first level of intelligence in a smart system.

3. Algorithms

There are different optimization algorithms that are evaluated in this work: (1) Global Optimization; (2) Segmented Optimization; (3) Neighbourhood Optimization, summarized in Tab. **1**.

The last two ones are suitable for distributed processing performing exploration, negotiation, characterized by self-* features. That means there are multiple instances operating on locally bounded data but with global objectives.

There is a set of observation variables: Strain ε, Stress σ, and the strain energy U finally computed, which is an objective variable, too. The optimization variable is the stiffness of elements ε, which is used to minimize the strain energy, stress, or strain. I.e., the optimization (objective) function is: $\min U(\varepsilon,\sigma)$ with $\varepsilon(e)$, $\sigma(e)$. .

Global Algorithm

do with $x \in \{\varepsilon,\sigma,U\}$
 $X:=0$; $\forall n\in\mathbb{N}$ do $X := X + x_n$
 $X := X/|\mathbb{N}|$
 $\forall n\in\mathbb{N}$ do
  $r_n := k_x(x_n/X, s_n)$
  $s_n := s_n * r_n$
until $|Err| < Err_0$
$k_U^1 :(q,s) \rightarrow$ if $q*w\in[r_0,r_1] \vee s\notin[s_0,s_1]$ then 1 else
 elsif $r*w<r_0$ then r_0 else r_1
$k_U^2 :(q,s) \rightarrow$ if $s\notin[s_0,s_1]$ then
 1 elsif $q*w\in[r_0,r_1]$ then $q*w$
 elsif $r*w<r_0$ then r_0 else r_1

Segmented Algorithm

do with $x \in \{\varepsilon,\sigma,U\}$
 $\forall \boldsymbol{\mathcal{S}}_i\in\mathbb{S}$ do
  $X_s:=0$;
  $\forall n\in\boldsymbol{\mathcal{S}}_i$ do $X_s := X_s + x_n$
  $X_{s,i} := X_{s,i} / |\boldsymbol{\mathcal{S}}_i|$
  $\forall n\in\boldsymbol{\mathcal{S}}_i$ do
   if $\neg\exists m\in\boldsymbol{\mathcal{S}}_j \mid m=n\wedge j\neq i$ then
    $r_n := k_x(x_n/X_{s,i}, s_n)$
   else
    $r_n := k_x(x_n/(X_{s,i}|X_{s,j}),s_n)$
   $s_n := s_n * r_n$
until $|Err| < Err_0$

Neighbourhood Algorithm

do
 $\forall\{n_i,n_j \in \mathbb{N} \mid i\neq j \wedge |pos(n_i)-pos(n_j)|=1\}$ do
  if $u_i < u_j \wedge s_i-\Delta s > s_0 \wedge s_j+\Delta s < s_1$ then
   $s_i := s_i - \Delta s$
   $s_j := s_j + \Delta s$
  end if
always

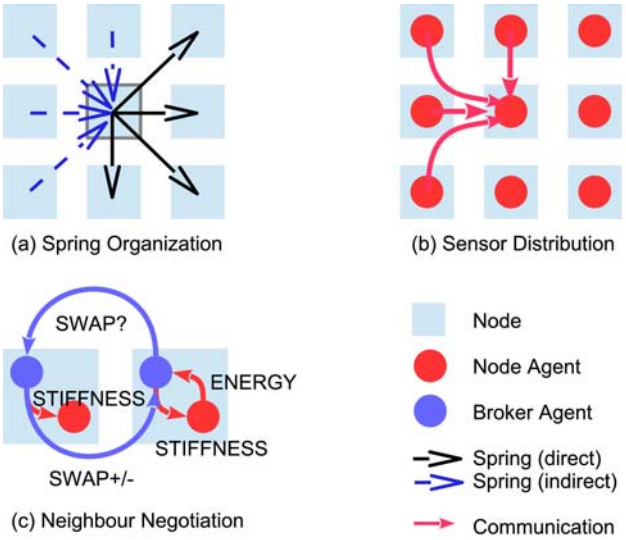
Table 1. Different optimization algorithms (simplified), with U: Strain energy; ε: Strain; σ: Stress; x_i: observation variable of i-th node (element), x: Some observation variable; X: Some accumulated observation variable; r_i: Optimization ratio parameter for i-th node, $|X|$: Cardinality of a set X, D: Some discretization function; s_i: Stiffness of i-th node/element, $[s_0,s_1]$: Limits of stiffness, $[r_0,r_1]$: Limits of optimization parameter, k_x: Some problem specific mapping and scaling function with weight factor w; $|pos_1\text{-}pos_2|=1$: Neighbourhood relationship of elements or segments

The basic principle of optimization uses a ratio parameter *r* based on the relation of observation variables (either ε, σ, or *U*) to a spatially extended ensemble value of this variable (mean). The ratio parameter is applied to the stiffness variables of elements of the material/structure. In Tab.**1** there are two different example functions k_U^1 and k_U^2 shown that compute the actual ratio parameter using the strain energy $U(=x)$. There is a set of all nodes: $\mathbb{N}$, a set of all segments grouping nodes: $\mathbb{S}$, and segments with a sub-set of nodes: $\boldsymbol{\mathcal{S}}\subseteq\mathbb{N}$. The segmented algorithm performs modification of elements partitioned in segments based on locally bounded data. The neighbourhood algorithm performs element modification between two neighbour nodes only (point-to-point).

4. Agents: The Computational Model

As already outlined, centralized sensor processing and actuator control is not applicable to the proposed smart structure and material model due to scaling and efficiency reasons. The resource constraints prohibit the deployment of conventional data processing and communication architectures. Additionally, for robustness and flexibility reasons it is desirable to decouple hardware and software entirely. The agent's computational and communication model provides such required decoupling. Agents are characterized by its loosely coupling to the underlying agent processing platform (APP), their capability to cooperate and co-ordinate, mobility (of agent processes), and finally to compose self-organizing and self-adaptive systems. The decomposition of a complex large-scale problem into multiple small low-complex entities is ideally reflected by the agent model. The behaviour model of the agents used in this work is reactive and is related with the Dynamic Activity-Transition Graph (DATG) model that can be implemented on various platforms **[5][6]**.

Agent Processing Platform: The choice of a suitable APP (details can be found in **[6]**) is actually still critical due to the hard resource constraints. The JAM JavaScript approach used in this work cannot actually scaled towards micro level (only macro level is supported) due to the JS Virtual Machine resource requirements (C>100MIPS, M>16MB). For micro level applications, a FORTH stack processor approach is available (AFVM, AgentFORTH), although the currently available sub-micro computers still not support this approach fully (but close enough as outlined in the next section, C>1MIPS, M>100kB). Both platforms are compatible on a meta operational level supporting the same agent behaviour and met programming model (AAPL **[5]**) that enables the combined deployment in heterogeneous networks.



Multi-agent System: The MAS is composed of different agents. Each node is equipped with a stationary (non mobile) node management agent performing sensor processing, distribution, and actuator control (here by controlling the stiffness of spring-like node interconnections). A broker agent performs modifications of the actuators, here the stiffness (see Fig. **2**).

Figure 2. Principle agent actions: (Top, left) Springs and sensors under node control (Top, right) Sensor data distribution to neighbour nodes by remote tuple access (Bottom) Stiffness negotiation and swapping between neighbour nodes by broker agents

5. Hardware Architectures and Technologies

The underlying hardware components depend on the geometrical scaling level: Micro (Material) or macro (Structure) level. The macro level includes robotic structures. The following discussion shows that the ICT and sensor parts are already existing, see Tab. **2**, enabling material integration **[7]**. The actuators modifying material or structure properties are still in an early technological state. In the simulation model used in this work parameterizable damped springs are assumed. Some polymers are suitable for this approach. There are different sensors: Strain gauge, displacement, temperature. Networks are arranged in 3D mesh-grids consisting of nodes connected with up to six neighbour nodes

The ICT on micro level assumes $1mm^3$ SoC computers, about 10MIPS, 100kB RAM, 100kB ROM, Serial Links; and on macro level computers of size $10cm^2$ are assumed, providing about 1000MIPS, 512MB RAM, 4GB ROM, WLAN; Ethernet, USB. The agent framework uses either a JavaScript VM (Macro Level) suitable for Embedded Computers, Mobile Devices, PC, Server, Cloud or the AgentFORTH Processor (Micro level) suitable for $1mm^2$ SoC Microchips. Communication is performed wired or wireless via serial links (electrical, optical, radio-wave), with about 100kB/s-1MB/s bandwidth.

Integrating computation in materials and structures requires a down-scaling of established algorithms, computer architectures, and co-ordination principles. A mobile computer equipped with an Intel *i5*-2520M and 4GB DRAM requires roughly A=150 (CPU) + 500 (DRAM) mm^2=650mm^2 chip area, delivering C=50000 MIPS computing power, but demanding about 35(CPU)+5(DRAM)=40W electrical power. Mid-scale computers, e.g., smart phones, are equipped with low-power devices, e.g., an ARM Cortex A9 delivering C=7500 MIPS (@ 1.5GHz), requiring only A=7mm^2 chip area and P=2W power. A **normalized computing power efficiency** of a computer (considering only the data processing unit) can be defined by $\varepsilon = C/(AP)$, and a relative down-scaling ratio factor is given by $s=\varepsilon_1/\varepsilon_2=C_1A_2P_2/(C_2A_1P_1)$. A scaling factor $\gg 1$ is desired. The down-scaling from an Intel *i5* to an ARM Cortex is expressed in a scaling factor of about 50. Material-integrated computing systems limit the size of a computer to roughly 1mm^2 chip area to reduce the mechanical impact of electronic components on the structure. An ATMEL *ATtiny* 20 micro-controller delivers C=12MIPS and requiring A=1mm^2 and P=20mW. Compared with an *i5* this gives a scaling factor of s=63. New trends in microelectronics pose 3D structuring of electronic devices (e.g., Micro Mote M3), increasing the computational power and memory storage multiple times. The functional layer of a silicon die has a width about 10μm (e.g., extracted by chip thinning), delivering up to 1000 MIPS/mm^2 or 250MBit/mm^3 assuming a simplified functional-interconnect-isolation layer sandwich structure.

Property \ Platform	**Michigan Micro Mote (M3) [4]**	**ELM System [4]**	**Atmel Tiny 20**	**Freescale KL03**	**ARM Cortex Smart Phone**
Processor	Arm Cortex M0	C8051F990 (SL)	AVR	Arm Cotex M0+	Arm Cortex A9
Clock	740kHz max.	32kHz	-	48MHz	1GHz
CPU Chip Area	0.1mm^2	9mm^2	1mm^2	4mm^2	7mm^2
RAM/ROM	3-4kB	0.5kB/2-8kB	-	2kB/40kB	512MB/4GB
Sensors	Temperature	-	-	-	Temperature, Light, Sound, Video, Acceleration, 3D Position, GPS, Magnetic, Air pressure
Communication	900MHz radio, optical	optical	electrical	-	3G/4G, WLAN, USB, Bluetooth, NFC
Harvester/Battery	Solar cell/Thin film	Solar cell/Coin	-	-	-
Power Consumption	70μW / CPU	160μW / CPU	20mW	3mW @ 48MHz	100mW avg., 2W peak @ 1.5GHz
Manufacturing process	180nm CMOS	-	-	-	40nm CMOS
Package	Wire bonded Silicon Stack	PCB	Single Chip	Single Chip	PCB
Computing Eff. ε	**150**	**0.02**	**0.6**	**4.0**	**0.53**

Table 2. Comparison of different ICT Hardware Platforms (Sub-micro computers) suitable for Smart Materials compared with a smart phone system

The M3 Micro Mote delivers the best computing efficiency, followed by the Freescale KL03 (but lacking energy, sensors, and communication modules).

6. Software Model

Multi-agent systems forming the distributed computational model offering co-operating agents. Agents are mobile processes that can migrate between network nodes of the material/structure and between inside and outside (Internet). Self-organizing and self-adapting systems (SoS) offer a divide&conquer decomposition of the global optimization problem in a distributed system with multiple processing instances operating each on locally bounded data. SoS is implemented by cooperating and co-ordinating agents. The JavaScript Agent Machine (JAM) is used for the processing of agents. JAM is implemented entirely in JavaScript and therefore portable; providing an extensive agent API with modules for shared database access (Tuple spaces, SQL), Digital Signal Processing (DSP), Machine Learning (ML). The ML module offers different models and algorithms: Decision Tree Learner, Artificial Neural Networks, and many more, integrated in JAM. Numeric frameworks are provided either by Matlab or are integrated in JAM. Used phys. simulation tools are: FEM: Abaqus, MBP: CANNON.

7. Multi-* Simulation

The design and technological implementation of smart adaptive materials is a challenge. Fundamental concepts has to be proven before any real system can be developed. Due to the strong coupling of sensing, reactive control, and information processing a multi-domain and multi-scale simulation has to be performed.

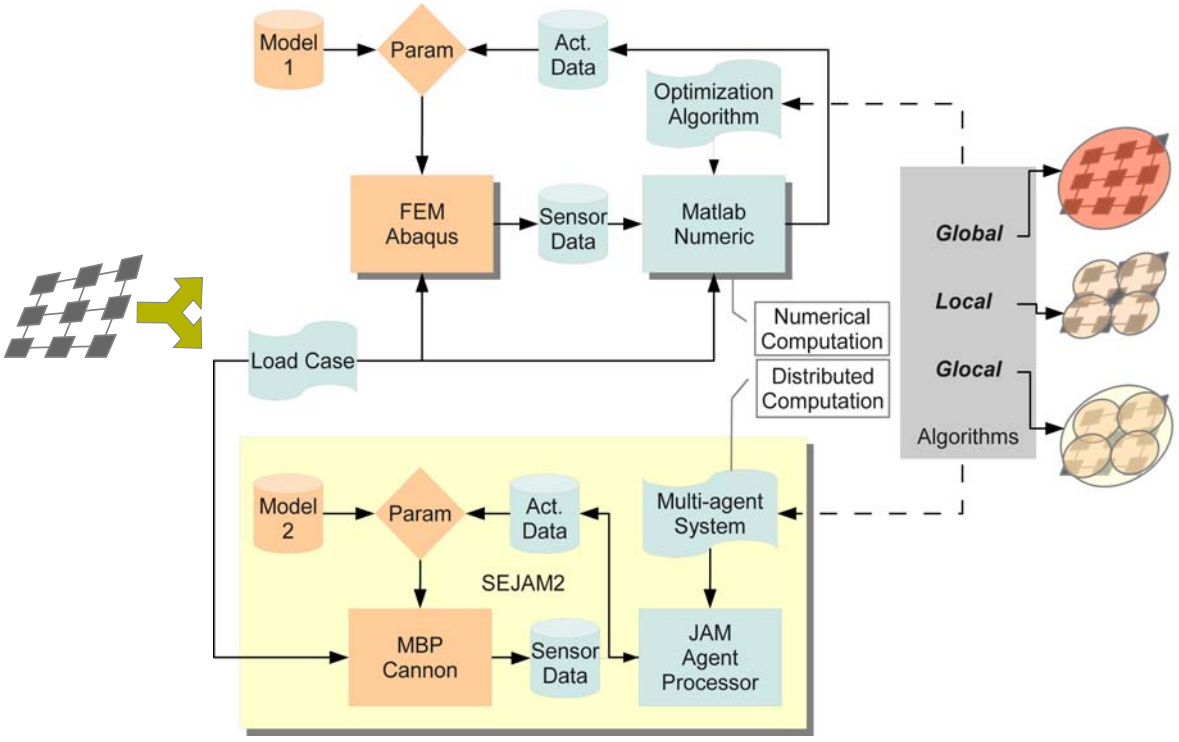


Figure 3. The entire simulation environment. (Top) FEM and Numeric using Abaqus and Matlab (Bottom) Multi-body Physics and MAS using the monolithic SEJAM2 simulator (Right) Different Optimization Algorithm Classes

Fig. **3** shows the entire simulation environment and the data flow consisting of: The Simulation Environment for JAM (SEJAM), a 3D multi-body physics engine CANNON, FEM analysis using Abaqus, and numerical processing with Matlab.

Two different mechanical models and solvers are used:

1. Finite Elements Mechanics (FEM); and
2. Multi-body Physics (MBP, based on coupled spring-mass networks).

The second model is closer to the proposed computational mesh-grid node network architecture and structures on macro level, whereby the first model is better related to materials on micro level. Based on these two mechanical models two different computational frameworks performing applying structure/material optimization are used.

FEM-NUM: Finite Element Analysis is used to develop and evaluate optimization algorithms performing. e.g., the minimization of a target function numerical, e.g., reducing the total strain energy, local strain, or inner force peaks. FEM analysis is stationary, i.e., no swinging or oscillation of the structure under test is considered, and a FEM simulation iteration ends with a stationary state of the structure. Algorithms operating on global (centralized) and local data (distributed) are investigated using FEM simulation and Matlab performing the reactive computational part of the algorithms.

MBP-MAS: Simulation of Multi Body Physics in combination with MAS is used to investigate and evaluate the computational and coordination model proposed in this work (details in **[6]**). In contrast to static FEM analysis the MBP simulation is dynamic and provides real-time resolution behaviour of structures. I.e., the MAS interacts with non-stationary states of the structures (swinging, oscillation, ..), too, which is closer to real-world interaction. The MAS implements the algorithms investigated in prior FEM analysis.

One major difference between the FEM-NUM and the MBP-MAS approach effects the sensor processing. The nearly real-time resolution MBP-MAS approach performs sensor distribution by MAS communication that introduces different time delays effecting the optimization algorithms and their outcome. Additionally, the MAS operates event-based leading to varying responses to a sensor stimulus as a result of structure dynamics.

8. Experimental Results

Two different use cases were evaluated: (1) A beam using the FEM-NUM approach; (2) A plate using the MBP-MAS approach. Some selected results are shown in Fig. **4** comparing the outcome of different observation variables (strain energy U and strain ε) for the the non-optimized and the optimized structure using different algorithms. In most cases an improvement about 30-40% can be achieved. The iterative global and segmented algorithms require less than 10 iterations to satisfy given error conditions (difference between target and actual state of the structure).

From a mechanical point of view, a reduction in total strain energy equals an increased structural stiffness. In practical applications of the concept, additional boundary conditions and side aspects need to be taken into account, like the effect of stiffness modification on ductility, damage tolerance or fatigue strength.

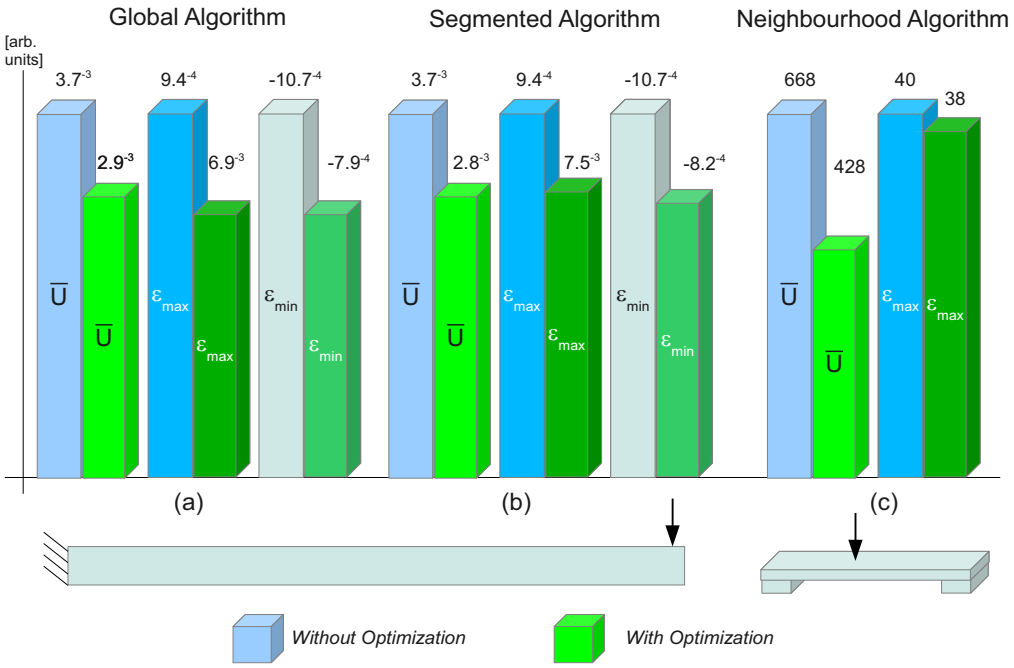


Figure 4. Simulation results show the optimization of strain energy U, maximum and minimum strain ε using different algorithms.(a,b): Beam, FEM+NUM; (c): Plate, MBP+MAS

Besides, additional optimization aims may be focussed on: Imagine e.g. a damaged structure in which a crack or hole leads to local stress concentrations - these could be alleviated by a stiffness adaptation aimed e.g. at levelling out stress and/or limiting stress gradients.

9. Conclusion

Combining materials posing adaptive properties (e.g., based on polymers) with self-organizing and self-adaptive computational agents can enable the design of smart materials and structures of the future. Among controllable material properties, the integration of computation within materials is an enabling technology and a challenge to be solved on multiple scales.

A multi-domain and multi-scale simulation framework offers a testbed for the design and evaluation of new optimization algorithms under computational, communication, and technical constraints with real-time resolution. Different proposed optimization algorithms could be evaluated and showed a significant minimization of strain and strain energy of structures under varying load conditions.

10. References

1. M A McEvoy, Nikolaus Correll, Materials science. Materials that couple sensing, actuation, computation, and communication, Science, 347(6228):1261689 (2015)
2. H. Janocha, Ed., *Adaptronics and Smart Structures*, 2nd ed. Springer (2007).
3. D. Lehmhus, S. Bosse, *Self-adaptive Smart Materials: A new Agent-based Approach*, doi: 10.3390/ecsa-3-S2005 (2016)
4. M. Choi, Y. Sui, I. H. Lee, R. Meredith, Y. Ma, G. Kim, D. Blaauw, Y. B. Gianchandani, T. Li, *Autonomous Microsystems for Downhole Applications: Design Challenges, Current State, and Initial*

*Test Result*s, doi:10.3390/s17102190 (2017)

5. S. Bosse, *Unified Distributed Computing and Co-ordination in Pervasive/Ubiquitous Networks with Mobile Multi-Agent Systems using a Modular and Portable Agent Code Processing Platform*, doi:10.1016/j.procs.2015.08.312 (2015).
6. S. Bosse, D. Lehnhus, *Towards Large-scale Material-integrated Computing: Self-Adaptive Materials and Agents*, doi: 10.1109/FAS-W.2017.123 (2017)
7. S. Bosse, D. Lehmhus, W. Lang, M. Busse (Ed.), Material-Integrated Intelligent Systems: Technology and Applications, Wiley, ISBN: 978-3-527-33606-7 (2018)