Jörg Kiefer and Eckehard Schnieder

Automatic partitioning and parameter identification for CAN–applications using genetic algorithms

Abstract. In this paper a method for dimensioning of field bus systems is presented using timed, coloured petri nets. A brief description of the test–environment which allows automatic parameter adaption, performance analysis, and application dependant comparison of field bus systems is given. A general structure is introduced allowing the simulation of different field bus systems. The stress is put on an application dependant optimization of the protocol parameter using genetic algorithms. These have to be adopted to the field bus protocol whereby the Controller Area Network (CAN) is taken for example. Using this approach, an automatic configuration of an individual CAN–application is possible leading to an optimal configuration. Real–time requirements of the application are transformed to maximum transmission times of the individual communication links. This leads to the possibility of automated parameter identification for the CAN.

Keywords. field bus, design, genetic algorithms, petri net, simulation, Controller Area Network (CAN)

1 Introduction

The use of automation systems in all fields of industry is increasing rapidly. All kinds of manufacturing systems are full of highly sophisticated technical equipment which is used for control, detection of machine states, logistical purposes and so on. During the last decade, the architecture of those systems changed from stand-alone solutions to open systems with strong communication links. This led to the requirement of a cheap communication channel providing data exchange with high reliability, and which is not application dependant. Field bus systems require only simple maintainance and their flexibility means that existing automation systems are easily expanded.

Nowadays, more than 100 different communication protocols are used in Europe with each of them having been developed for a specific purpose. The designer of a communication network has an unsolvable task choosing the right bus system, adopting the application to the different types of telegrams, and adjusting the bus parameter. The result of this will always be very dependant on the knowledge of the engineer.

Therefore, an approach is required supporting all different steps of the design. Only the requirements of the process are taken. All further steps (e. g.: allocation of the functional units to processors, adoption of the protocol parameter) have to be done automatically. This even allows an easy comparison of different field bus systems which are optimised independantly.

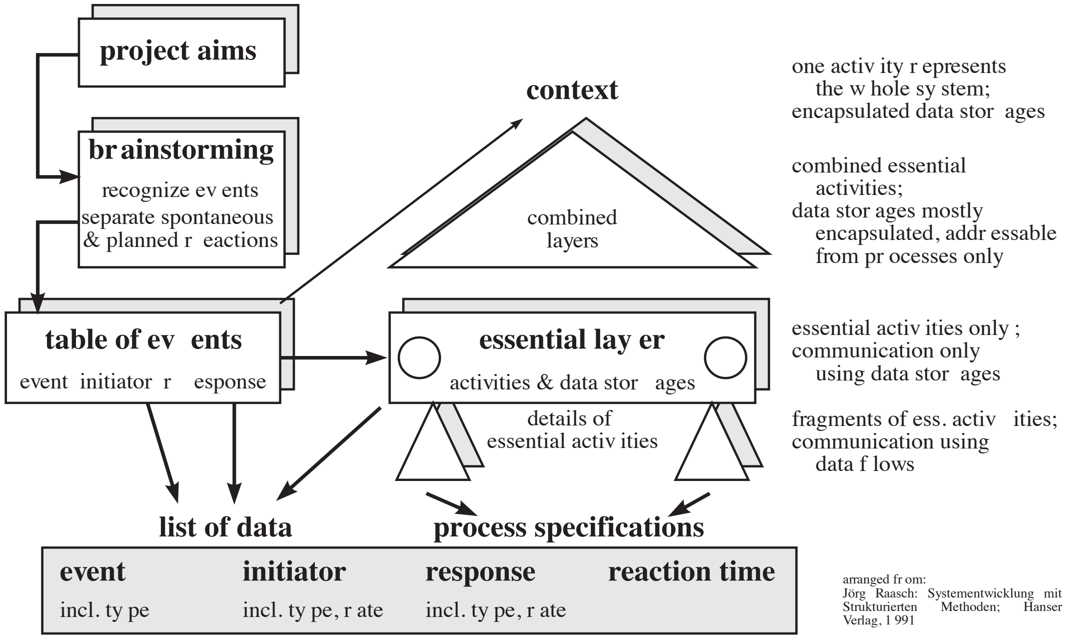
2 General approach and test–environment

The time behaviour of a communication network is always very dependant on the application itself. Therefore, the modelling of the application is one of the most critical tasks in performance evaluation. It is well known that an error in setting up the input data will lead to simulation results which do not have any relevance to the automation system. A method has to be found ensuring precise input data for the

---

[0] Technische Universität Braunschweig, Institut für Regelungs– und Automatisierungstechnik, Langer Kamp 8, D–38106 Braunschweig, Tel.: + 49 (531) 391–3317, Fax: –5197
Since June 1st, 1995, J. Kiefer is employed by Siemens AG, Transportation Systems, R&D Innovative Railway Signalling, Ackerstrasse 22, D–38126 Braunschweig, Tel.: + 49 (531) 226–3656, Fax: –4673

simulation tool. A straight forward approach can be achived when using the language which is mainly used for the specification of the process itself.



arranged from: Jörg Raasch: Systementwicklung mit Strukturierten Methoden; Hanser Verlag, 1991

Figure 1: The process of structured analysis and design of an automation system

The whole method is based on the high level process description language Structured Analysis (SA) and its real–time module (SA/RT). It can be applied to most applications without respecting special needs in case of a local area network. This leads to the following steps in design engineering resulting in a data set which can be used for our purposes. The design processes are shown in Figure 1 but will not be discussed here. For detailed explanation, the reader should refer to [1] and [11].

The set of input data consists of all information which can be gained from the design process without having any additional design steps due to the use of a communication network. External events including its initiators and system responses are deduced as well as the time criteria.

From this formal description, the individual functions can be allocated to hardware units with respect to the real–time constraints, and the overall communication load in the process. Even this algorithm is applicable to any communication system leading to a requirement specification of a communication network with its communication channels and its real–time constraints.

In the next step, the different parameters for the individual channels have to be adjusted. When using the Controller Area Network, this can either be done by using a CAL–service or setting the parameters for identifier (ID) and remote transmission request (RTR) manually. The last approach is mainly used for time critical applications in automation engineering requiring minimum transmission times. Nevertheless, the number of different combinations is enormous and cannot be tested completely. Therefore, genetic algorithms are used for optimization of the parameters. Together with an abstract simulation of the process (data can be gained from SA/RT modelling) and a CAN–module, the parameters are adopted off–line without using the process equipment itself (important for safety critical processes) or implementing the CAN in hardware.

For the dynamic performance analysis the temporal behaviour is most important. This has to be considered when modelling a field bus systems. The request for a stochastical simulation can be deduced from the field bus systems, as well. On the physical transmission line errors might occur which are due to influences from magnetic or electric fields. This phenomena is normally described by a fixed probability. For simulation of high system loads this effect has to be taken into account.

Regarding communication networks, we can see that most of the bus stations are physically equal con-

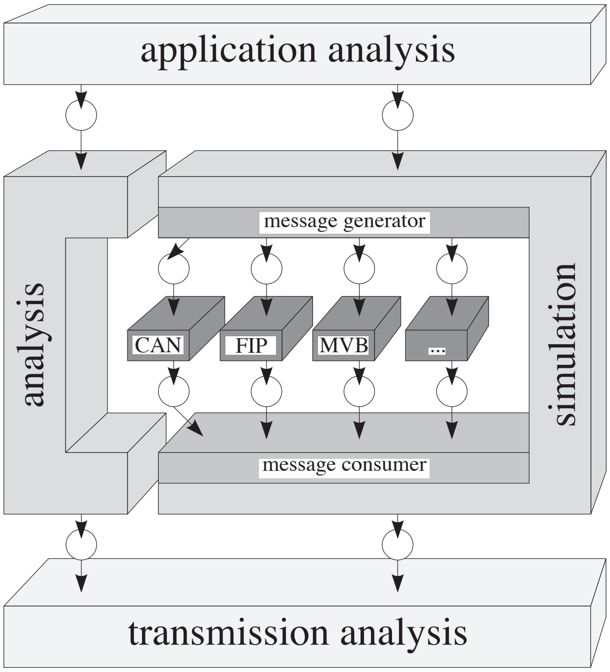


Figure 2: Test environment for bus examinations

necting to other process units. For this purpose, a simulation allowing aggregation of different units is used.

Last but not least, an analysis unit allowing a formal verification of some of the transmission criteria should be implemented as well. For this reason, the field bus models as well as the synthetic message generator should use a formal language allowing a mathematical treatment of the model. This aspect is needed in automation systems where a safety proof has to be done. In this case, maximum transmission times have to be calculated.

A schematic view of the whole system can be taken from Figure 2. The units described can easily be seen as well as the different field bus modules.

All the simulation's requirements are met by coloured, stochastical, and hierarchical petri nets. Details about the simulation can be taken from [6, 5].

3 Parameter optimization for the CAN

3.1 Parameters of the CAN-protocol

The different field bus systems have a large variety of protocol parameters. From the mathematical point of view, they are either a permutation of the data links or a combination of attributes that can be applied to the data.

For time–critical applications, it is most useful to set the parameter of the CAN-protocol manually not using the CAL–services. Two parameters are gained:

1. The **identifier** of every data is a permutation of the number of generated variables in the network.
2. At the other hand side, the use of the **remote transmission request** for data transmission is a combination of two possibilities (RTR = 0/1) for every channel.

The RTR is hardly used for time–critical applications because the amount of transmitted data is higher than in event–orientated mode. Therfore, it will be neglected for the further examinations.

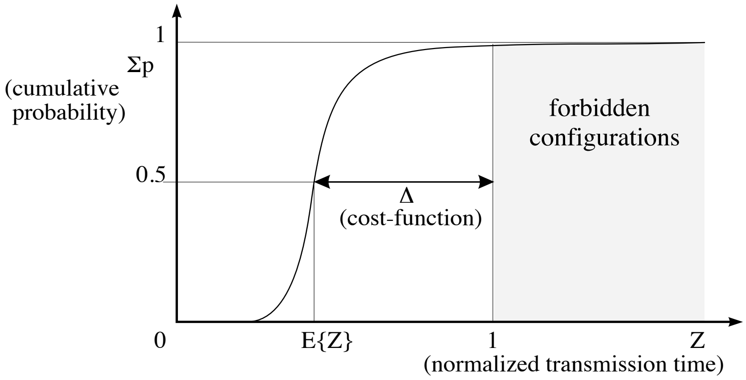


Figure 3: Scematic view of the cost–function

3.2 Cost–function for the optimization

For the judgement of the different combinations of parameter, a cost–function is implemented. It has to be taken into account that there should not be any aspects that are network dependant. This allows the comparison of different bus systems, and leads to functions that use process requirements, only. Two different aspects have to be taken into account:

1. Some processes have a maximum response time for the control system. This leads to a rejection limit for a data path. There might be a number of different functions and several data transmissions in one data path. Only the overall time can be compared with the requirement. This criterion can be evaluated by a statistic test leading to the valid configurations.
2. In addition, a comparison of the valid configurations is necessary leading to the optimal configuration. In this case, the difference of the average transmission time and the required time is taken. Figure 3 shows this criterion wherby the abscisse shows the simulated transmission time normalized by the maximum transmission time.

The cost–function for the judgement of the different configurations uses the maximum time–limits of the process for determination of configurations fulfilling the real–time demands. At the other hand side, the valid configurations are ordered with respect to robustness of transmission and the absolute bus–load.

3.3 Choice of an optimization algorithm

From the mathematical point of view, the adaption of the parameter can be seen as a problem of non–linear optimization. Many algorithms have been developped using the gradient of the function to determine the optimum solution. These cannot be used for our purposes because of two reasons: First, the gradient is very much discontinuous, second, there is no natural order of the input combinations. The last point is required by all search algorithms so that none of them is applicable.

Genetic algorithms [3] do not have any requirements concerning the input data. Therefore, they are best suited for problems that can be coded by integers.

3.4 Genetic Algorithms

3.4.1 General aspects

Genetic algorithms have been derived by examination of the principles of evolution. Therefore, they use three different operators:

- **Mutation** causes a random change in the information of an individual. It is a nondirective operator which prevents finding a local optimum.

- **Crossover** exchanges some of the informations of two individuals.
- **Selection** determinates those individuals that will be taken for the next generation. This is the directive part of the algorithm. Those characteristics having the best cost–function are preferred for recombination.

The algorithm combines parallel (different individuals) and serial (several generations) search strategies.

Nassi–Schneiderman diagram — Genetic algorithm

```
choose a suited coding
initialize the first generation (random)
    evaluate the fitness of all individuals
    select pairs in accordance to their fitness
    generate new individuals by crossover of the actual individuals
    mutate some of the new individuals
    replace parts of the actual generation by their children
repeat until abortion criterion has been reached
```

3.4.2 Coding and genetic operators

Choosing the right coding is one of the most difficult parts when applying genetic algorithms. In most cases, the choice of a coding directly leads to the related genetic operators for mutation and crossover.

Therefore, the different codings need to be classified. This allows an easy choice of the right coding with regards to the bus system's requirement. It has been found that three aspects can fulfill this task:

- **absolute position** of a characteristic in the string
- **relative position** towards another characteristic
- **direct sequence** of two characteristics

The classification of several codings for permutations is shown in Table 1. A formal discription of the different codings can be taken from [8].

Finally, we have to look at the requirements of the CAN–protocol: The identifiers of the CAN are directly related to the priority of the data. Therefore, the relative order of the different data links is important (relative position). The absolute position is not really required but this is not an independant value (permutation). Direct sequence is not required at all. This leads to the following constraints:

absolute position:	average (∘)
relative position:	high (+)
direct sequence:	low (−)

The ordinal–coding has been chosen for the CAN–protocol. At the same time, the genetic operators are fixed. They will not be mentioned in detail because this would require a formal introduction of the coding, as well.

Table 1: Classification of codings

codings	absolute position	relative position	direct sequence
vektor–coding			
adjacency–coding	∘	– –	– –
ordinal–coding	∘	+	–
path–coding			
cycle crossover	∘	–	–
order crossover	–	∘	∘
order–based crossover	∘	+ +	∘
partially–mapped crossover	+	+	+
matrix–coding			
precendence–coding	∘	+ +	–
Seniw–coding	+	∘	+
Hamaifar/Guan–coding	+	∘	+

4 Example: container terminal

A reduced scale model of a container terminal has been built at the institute for research purposes. The complete system consists of three travelling bridges, two loading robots, two gravity roller tables and a control center. The first mentioned devices are equipped with motors for the positioning of the axis, spreader and safety equipment (e. g. emergency stop transducers). The control center consist of a programmable logic control (PLC) for control, diagnosis and supervision. The configuration is shown in figure 4. A development task was to link the communication of all components via a field bus system. Due to the availability of PLC components the CAN–protocol got first choice and had to be examined concerning its capability for this application.

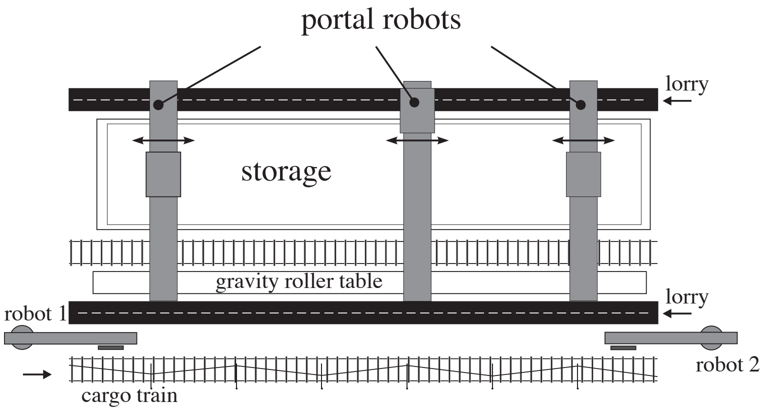


Figure 4: Scematic view of the container terminal

The different function units have already been allocated because of the complexity of the control system. A complete description of the process data for the allocation would not fit into this paper.

4.1 Requirements and functional description

Figure 5 shows the data flow in the application. It can easily be seen that this is a cascade control whereby the inner loops (1, 2) are operational and the outer loops (3 – 5) serve for error detection.

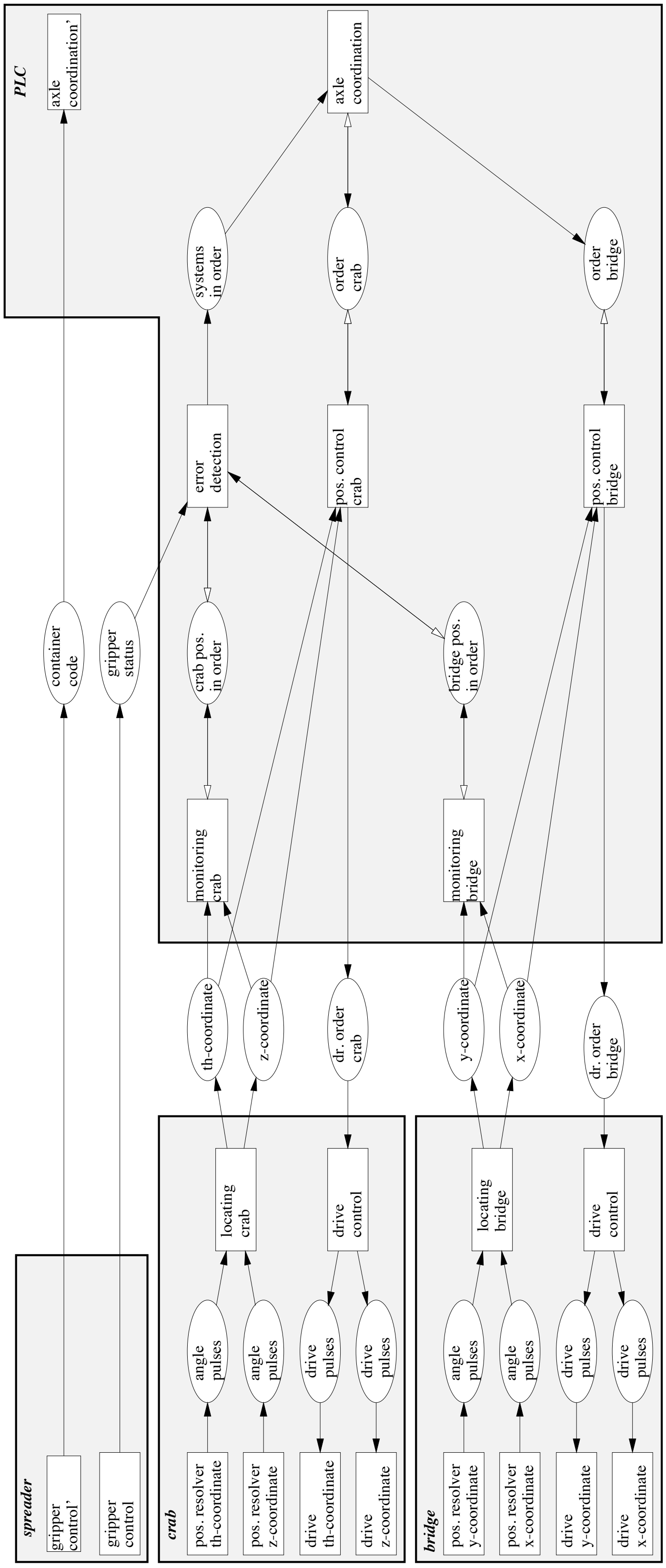


Figure 5: Functional model of the container terminal

Table 2: Table of time–critical events

no	event	trigger	response	max. time
1	drive control for crab	z–/θ–coordinate	drive pulses crab	$t_{max} = 19$ ms
2	drive control for bridge	x–/y–coordinate	drive pulses bridge	$t_{max} = 19$ ms
3	error in crab position	crab coordinates	drive pulses crab	$t_{max} = 48$ ms
4	error in bridge position	bridge coordinates	drive pulses bridge	$t_{max} = 49$ ms
5	error in gripper control	gripper status	drive pulses bridge	$t_{max} = 50$ ms

Due to a lack of calculation capacities on the controllers, it is not possible to implement the inner control loop on the moving parts. Table 2 gives the process requirements for the optimization. It has to be considered that every function needs time for its task.

4.2 Convergence and robustness

For better understanding of the process, all possible configurations have been simulated whereby only the timed critical links has to be looked at. It can be seen that about 65% of the configurations are valid. In the other cases, one of the error transmissions channels fails. This is due to the fact that all coordinates and the container code try to send at the same time. In that case, the identifier have to be adopted correctly not leading to a process failure.

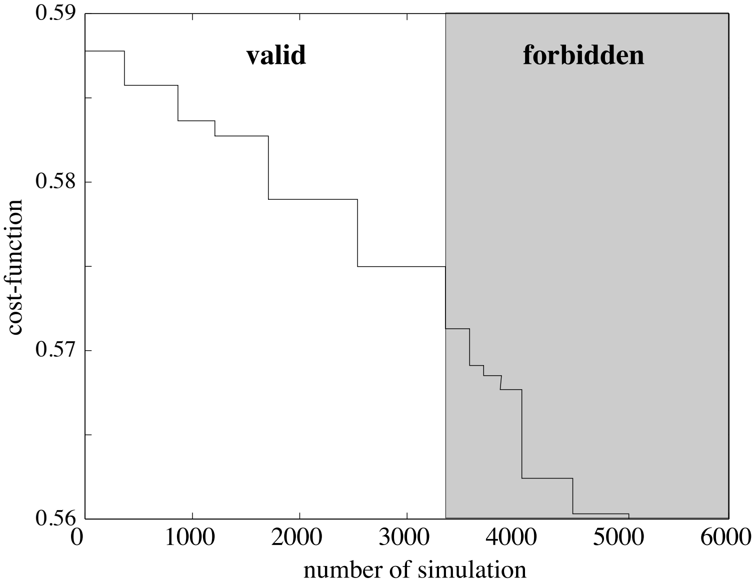


Figure 6: Complete simulation for the container terminal

It can be seen, that there are 13 steps, only. This is due to the fact, that the identifier can be exchanged as long as they belong to the same data path. They will always be executed sequentially not disrupting each other.

In Figure 7 the mean of 50 simulations has been taken. Though the variation of the parameter, the optimum has always been found; only the average is slightly influenced. This robustness leads to good results even with a more critical process. A rough adaptation of the algorithm's parameters is sufficient for the use with field bus systems

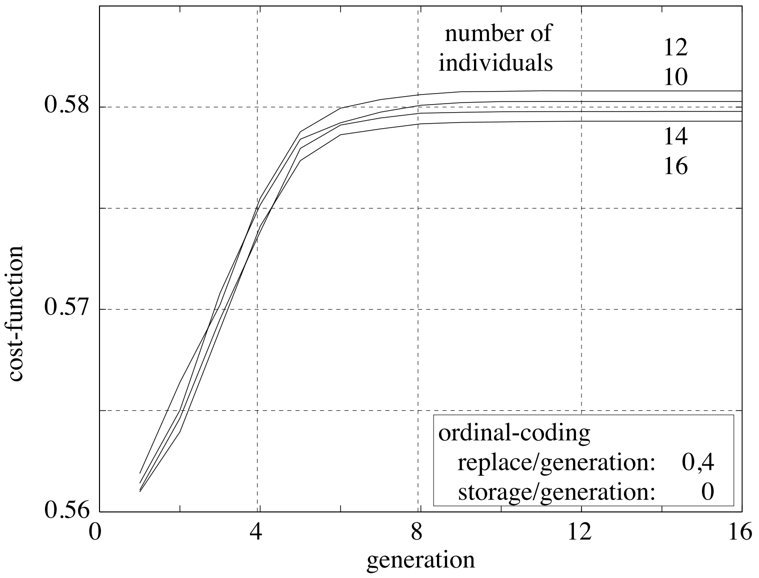


Figure 7: Convergence of the algorithm

4.3 Result of the optimization

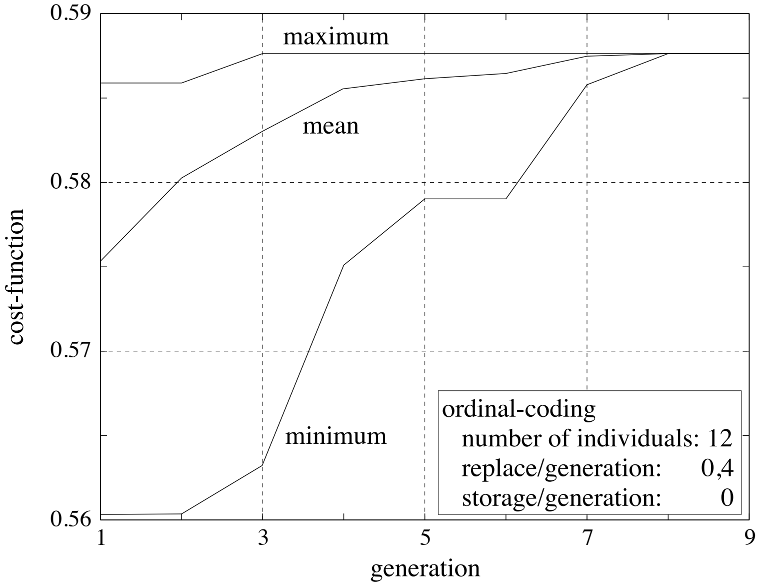


Figure 8: Convergence with ordinal–coding

Finally, the result of a typical simulation run is shown (Fig. 8). The three graphs show the characteristical values of the actual generation. The optimum is found very early which is due to the large number of good solutions. Nevertheless, the algorithm searches the whole range. The maximum reaction times for the different channels are fulfilled (transmission rate: 1 MBd, bit–error rate: 0.001). They can be taken from Table 3.

Table 3: Simulation results for optimised solution

no	event	simulated time	maximum time
1	drive control for crab	$t = 11.47$ ms	$t_{max} = 19$ ms
2	drive control for bridge	$t = 11.53$ ms	$t_{max} = 19$ ms
3	error in crab position	$t = 42.47$ ms	$t_{max} = 48$ ms
4	error in bridge position	$t = 44.39$ ms	$t_{max} = 49$ ms
5	error in gripper control	$t = 41.39$ ms	$t_{max} = 50$ ms

5 Conclusions

- Genetic algorithms can be used for automatic parameter optimization of field bus systems.
- The choice of the coding can easily be done by matching the protocol demands and the possibilities of the coding.
- The large number of good solutions in a network and the robustness of the genetic algorithms lead to very quick results.
- Petri net simulation of the field bus system allows to adopt all system parameter without implementing neither the network nor the process (important for safety critical processes).
- Design and implementation of field bus systems can be computer–aided without any additional system analysis. The approach is not application dependant.

References

[1] L. Erdmann, A. G. Schielke, and E. Schnieder. Referenzmodell zur Strukturierung von Leitsystemen. *at (Automatisierungstechnik)*, 42(5):187–197, 1994.

[2] K. Etschberger, editor. *Controller Area Network: Grundlagen, Protokolle, Bausteine, Anwendungen.* Carl Hanser Verlag, München, Wien, 1994.

[3] J. Holland. *Adaptation in natural and artificial systems.* MIT Press, Cambridge, 1992.

[4] J. Kiefer and E. Schnieder. Analysis of field bus systems for high speed trains. In *Theory and Application of Advanced Technology*, pages 1056–1061, Tianjin, China, August 1994. 7th IFAC/IFORS Symposium on Transportation Systems, International Federation of Automatic Control.

[5] J. Kiefer and E. Schnieder. Petri net modelling of field bus systems. In J. Billington and M. Diaz, editors, *Petri Nets applied to Protocols, Proceedings of a Workshop of the 16th International Conference on Application and Theory of Petri Nets*, pages 97–109, Torino, 1995.

[6] J. Kiefer and E. Schnieder. Simulation of field bus systems for design and parameter optimization. In *IASTED: MS'95*, Pittsburgh, USA, April 1995. IASTED International Conference: Modelling and Simulation.

[7] Meta Software Corporation, Cambridge, MA. *Design/CPN Reference Manual*, 1993.

[8] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, 1992.

[9] Philips Semiconductor, Hamburg. *CAN–Specification*, 2.0 edition, 1991.

[10] I. Rechenberg. *Evolutionsstrategie.* Friedrich Frommann Verlag, Stuttgart, 1973.

[11] P. T. Ward and S. J. Mellor. *Structured development for real–time systems.* Kluwer Academic Publishers, London, 1988.