ISSN 2229-5518

Frame Communication Module for Bin-Packing Algorithms for Distributed Embedded Systems

Mr. Anand, Dr. S.Ravi, Mrs.V.Kavitha and Dr. Uma Rajaram

Abstract--- Embedded real-time systems must satisfy not only logical functional requirements, but also para-functional properties such as timeliness, Quality of Service (QoS) and reliability. The proposed scheme describes an automated schedulability analysis, and generates glue code to integrate the final runtime executable for the system. Its extensive glue code generation capabilities include the ability to insert inter-processor communications code at arbitrary software boundaries. The objective of this deployment is to minimize hardware requirements while satisfying the timing constraints of the software. The classical approach to addressing this problem is to use bin-packing techniques. A bin-packing algorithm is proposed to exploit the capability of partitioning software modules into smaller pieces which exhibits that number of bins required can be reduced. In this paper, we investigate how to assign signals to periodic frames with the objective function to minimize the network bandwidth requirement while not violating specified deadlines. This problem is NP-hard, but can for most typical applications be solved efficiently by using simple heuristics. The effectiveness of our algorithm is demonstrated by applying it to signal sets derived from automotive applications for a CAN based system and for the newly developed, low cost and low speed, Local Interconnect Network (LIN). The results can be of great use in cost sensitive embedded systems such as car control systems, where the used hardware, communication networks and nodes (typically micro-controllers), have to be highly utilized to keep the production cost at a minimum level.

Index Terms--- Bin packing Algorithm, CAN, Embedded system, Interprocess communication, Local interconnection Network, Quality of service, Schedule ability analysis.

1 INTRODUCTION

Embedded real-time systems are tightly coupled with physical world. This tight coupling imposes para-functional requirements (such as timeliness, jitter, fault-tolerance, and security) that go beyond functional (logical) behaviors. Each functional behavior and parafunctional behavior has to be captured, where each view focuses on a single concern enabling domains such as signal processing, control systems, real-time systems, and a fault tolerance has to be integrated. For instance, software-defined radio model presents a functional view to the signal processing engineers to compose the mathematical transformation of signals in order to modulate/demodulate them.

(i) A domain models the Fault-Tolerance (FT) structure can be modified by the FT expert.

(ii) The FT expert defines which functional modules should be replicated to tolerate faults, how to synchronize their execution such that their internal state does not deviate from one another, and how to ensure failure independence (e.g., by deploying them on different processors).

(iii) In this domain, the Real-Time (RT) Systems expert can monitor the rate of execution of a component relates to that of another and timing relationships among components.

The integration exposes the impact of changes in one domain with the others. For instance, the addition of a new component in the functional domain is propagated to all the other views (Timing, Fault-Tolerance, and Deployment), the replication of modules is propagated to the Deployment view to be able to deploy the new replicas, and the timing relationships among components are propagated to the deployment view to evaluate whether the available processors have enough cycles to execute the software within their timing constraints.

The proposed scheme focuses on the automatic assignment of software modules (or tasks) to hardware nodes. The traditional approach to allocate software modules to hardware nodes is to use a bin-packing scheme. It exhibits the flexibility to "partition" a software module across two or more processors can lead to a reduction in the number of processors needed. Diametrically opposite to this the signals are packed in as few frames as possible and there exists two different sizes of frames such as frames that carry different amounts of data and have different

Mr.M.Anand is a Research scholar, St. Peter's University, Chennai, and also Asst.Prof. in Dr.M.G.R Educational and Research Institute University, Chennai-95, India

Dr.S.Ravi, Professor and Head, Department of ECE, Dr. M.G.R Educational and Research Institute University, Chennai-95, India ravi_mls@yahoo.com

Ms.Kavitha, Research scholar, Dr. M.G.R Educational and Research Institute University, Chennai-95, India

Dr.Umarajaram, Dean (Engineering & Technology), Dr.M.G.R. Educational and Research Institute University, Chennai-95, India

transmission time. If the signals have about the same deadline, it would be beneficial to send them in one large frame. Because the smallest deadline in each frame determines the period time of the frame, and thus, the bandwidth utilization would become less than packing all signals into a large frame. To assign signals to frames is difficult since,

(i) The signals are asynchronous (i.e., the different signals are available for the communication subsystem at non synchronized times)

(ii) Many protocols for embedded systems allow different frame sizes with different transmission times, e.g., in a CAN-based system, the data is transmitted in frames containing between 0 and 8 bytes of data.

1.1 Our Approach

The allocation scheme, models the software as a directed acyclic graph of modules that communicate through messages. Modules are characterized as consuming CPU cycles (periodically) and messages as consuming bandwidth (bits per second). A second graph is used to represent the hardware architecture. In this graph, the nodes are processors and networks, and the links represent connectivity among them by assuming only simple hardware architecture graphs such as bus based or switch-based networks.) Processors are considered to provide a processing capacity expressed in cycles per second and networks are considered to provide a communication capacity in bits per second. This problem can be addressed by Earliest Deadline First (EDF) scheduling algorithm [10] is used. When an object is partitioned and deployed on two (or more) processors, any timing constraint that applies to the object must now span those processors. An "inflation factor" to account for this overhead can be added later if necessary. The ideal objective of the packing problem is then to use the minimum number of processors and network links to deploy a software graph. A extension to bin-packing algorithms which allows partitioning of software graphs to minimize the number of processors needed to run the system while attempting to reduce the network links needed (due to messages across processors) as much as possible.

A finite set of signals for each node, where each signal is characterized by a deadline and a size with finite number of different sized types of frames with different transmission times. To devise a scheme for mapping of signals to periodic frames, which will minimize the bandwidth utilization of the communication network such that all of the signals are uniquely assigned to frames and that the frames are globally schedulable. When comparing different packing alternatives, the metric was chosen as utilization measure for a frame as the transmission time divided by the deadline of the frame, and consequently the utilisation measure for the network as the sum of utilisation measures for all frames. In the scheduling phase, periods are determined based on deadlines, frame transmission time, and the scheduling method used. A straightforward solution is to transmit each frame with a period time that is equal to half of the deadline then the deadline requirement for each frame will be fulfilled, but possibilities exist [13]. Thus for a set of frames where each frame has a period time and a transmission time, on scheduling analysis has to be made and several mature techniques for scheduling analysis of periodic frames for different protocols exist, including the technique developed for the CAN-bus [9].

The problem addressed was similar to the task allocation and scheduling problem that has been studied by many Researchers [9][11]. The main difference is that most often in task allocation, a system with a finite set of nodes is given while in this scheme considers have non-finite The multimedia sector poses problem with slightly more complex since they handle different kinds of resources like, disk, CPU and network resources. The contributions of this scheme are that,

i. Formulate the packing problem.
ii. Show that the packing problem is NP-hard
iii. Present a simple heuristic for frame packing that we show is very effective.
iv. Demonstrate the effectiveness of the algorithm on realistic sized problems derived from the automotive industry.

1.2 Related Work

Multiple research efforts have been conducted relating to deployment of software modules to hardware. In [1], propose a period-based load partitioning scheme that minimizes a metric called the system hazard, which is defined as the maximum response time among all tasks. These tasks are real-time tasks with precedence constraints and dependent execution times. A branch and bound algorithm was developed to cluster hardware nodes and software modules with the objective of performing an hierarchical allocation that minimizes the system hazard. In contrast, our work does not minimize the system hazard but the number of processors needed for a specific software graph. In [11] presents a tool to allocate real-time tasks to

processors. Their tasks are modeled as independent tasks and a fixed-priority scheduler with RMS [9]. A branch and bound algorithm is used to find a feasible allocation. In our scheme, we allow components to have communication dependencies.

In [13] develops a simulated annealing algorithm for real-time task allocation. Tasks communicate with each other using a token protocol and the task's deadlines are modified to take into account the delay of such communication. [2] response time analysis was used to calculate the "energy" component of the algorithm. In our case, we assume that the deadlines have already been adjusted for any delays due to dependencies and focus on the allocation of software to processors and messages to networks. Related to bin-packing, an algorithm to allocate divisible objects to bins was developed in [4]. However, the size of the objects are restricted to be in a special divisible sequence, where the size of object is divisible by the size of object, which in turn is divisible by the size of object and so on and so forth. Such a restriction makes this approach difficult to apply in practice. Our work does not have this restriction and allows components to be partitioned into any size. In the networking arena, [12] presents a binpacking algorithm to fit packets into a TDMA network. These packets can be fragmented but the fragmentation adds an overhead factor representing the additional bits that need to be sent. The purpose of the algorithm is to pack network packets compactly, avoiding fragmentation as much as possible. Even though this work also deals with object partitioning, in ourcase, we formally analyze the improvements on the worst-case performance of bin-packing algorithms due to partitioning, assuming that such partition does not impose any overhead.

2. BIN-PACKING ALGORITHMS

This scheme describes a deployment algorithm which is based on a new paradigm that may "partition" software modules into two or more pieces. In practice, partitioning pieces will lead to communication code insertion at the partitioning points. The communications code required to connect a partitioned object can be automatically generated. Also, both functional and para-functional behaviors can be verified with the assistance of built-in or external analyses. Hence, object partitioning to reduce the number of processors ("bins") is a very desirable objective. The proposed partitioning algorithm is derived from the traditional bin-packing algorithms. Bin-packing algorithms try to solve the problem of packing a set of objects into the minimum set of bins. In the basic version, the maximum size of each object is 1, and each bin is of size 1. A bin can be filled with objects until the total size of all the objects in the bin sum up to 1. Objects cannot be partitioned and must be allocated as whole. Previous research [8] has already shown that finding the minimum number of bins to pack a set of objects (namely the bin-packing problem) is NP-complete. However, multiple near-optimal algorithms of polynomial complexity have been designed. The most popular and best understood is Best Fit Decreasing (BFD). This algorithm orders the objects in decreasing order of size and the bins in increasing order of available space. BFD then tries the allocation of the first object in each of the bins in order. This scheme has two attributes. One, it assigns the largest objects first when it is more likely to find a larger gap. Two, BFD seeks to leave the minimum gap possible by assigning the largest object to the smallest gap that accommodates it. This second attribute is the basis of the less complex algorithm called Best Fit (BF). Other less complex algorithms such as First Fit and First Fit Decreasing have also been used for bin-packing. First Fit (FF) does not order the bins or the objects, and assigns an object to the first bin where it fits. First Fit Decreasing (FFD) is an improvement on FF where the objects are ordered in decreasing order to reduce the gaps in the processors. This scheme will not use these two algorithms because the worst-case performance of BFD is better than FF In the analysis of these algorithms, use XFD when the properties discussed apply to both BFD and FFD, and XF when they refer to either BF or FF. The proposed framework is designed to support distributed real time systems, where software modules on one processor may communicate with modules on other processors. If such communicating modules can be assigned to the same processor (as a composite one), their communication requirements across a network will be minimized or even eliminated. However, not all modules that communicate with each other may fit into the same processor. Furthermore, due to the (recursive) encapsulation of software components into composite ones, each component can became potentially large in size, to the extent that it may not fit into one processor. The binpacking scheme will therefore fail for such large objects. It avoids these problems by allowing objects to be partitioned (or split) into smaller objects down to individual elementary components by employing the following assumptions, first, each bin is assumed to be equal to 1 and secondly, each object ranges in size from 0 to 1. Finally, if an object is partitioned, the sum of its parts equals the size of the original object.

2.1 BFD WITH PARTITIONING (PBFD)

It refers the multiple objectives as a BFD packing problem with partitioning (PBFD). Mathematically, each software component can be considered to be a node in a graph 'G' with an edge added between two nodes if the two nodes represented by these nodes communicate. Then, each connected graph component is treated as a composite object that can be partitioned and assigned by the bin-packing scheme. In classical bin-packing, bins are of unit size. One starts with (zero bins or) 1 bin and additional empty bins are added when necessary. Adding bins on demand can lead to unnecessary and premature partitioning of objects. By adding bins on demand, start with a single bin b1 and allocate object 'w' of size. At this point, by considering object 'x' and decide that it does not fit in bin b1 and to split the component, in halves x1 and x2 putting them back in the list ({x_1, x_2, y, z}). Now, the next object x_1 can be assigned to bin b_1. Assuming that no other object can be partitioned and a new bin b_2 to be continued with the deployment. This bin can now hold the rest of the objects in the list ({x2, y, z}), leading to an assignment as follows:

b1 = {w, x1}, b2 = {x2, y, z}. Now, if instead of allocating bins on demand, two bins b_1 and b_2.

Next, we can allocate 'w' and 'x' to each of these bins. Then, allocate objects y and z to bins b_1 and b_2 leading to the assignment: b_1 = {w, y}, b_2 = {x, z}. The resulting network bandwidth requirement is zero given that no object. It turns out that a higher ratio can be achieved for lists that can be packed in two or three bins. This observation long made by bin-packing studies (e.g. [8]) also leads us to focus on lists of a large number of objects was partitioned.

1) **Early Partitioning:** This scheme partitions an object immediately if it does not fit into any of the pre-allocated bins.

2) **Late Partitioning:** This scheme defers the partitioning of the objects that do not fit into any of the bins until it has done deploying all the objects that can be deployed without partitioning.

This algorithm ensures that by deploying the largest number of objects without partitioning to try to minimize the penalty of the partition, i.e. communication between processors. It must be noted that that by enabling to accommodate all the objects in the pre-allocated bins, then our algorithms behave like vanilla BFD. The PBFD algorithm can be summarized as follows.

(i) The allocation order of BFD is following the bins by non-decreasing order of gaps and try the deployment of composites in decreasing order of size.

(ii) If partitioning is needed (by the partitioning scheme in use) the selected composite is partitioned into two parts that can be fit into the largest gaps5 available. If this partitioning is possible, these parts are returned to the list of objects to be deployed and the list is reordered (in descending order of object/piece sizes).

The object is partitioned, but instead of being deployed, the pieces are added to the list of objects to be deployed. If, on the other hand, the composite cannot be partitioned into parts that fit into any gap, then a new bin is added, and the unpartitioned composite is assigned to the new bin. Finally, the deployment continues until all objects have been assigned.

3. PROBLEM STATEMENT

3.1 System model

We assume a distributed system consisting of a set of nodes interconnected via a communication network. The Communication protocol is assumed to be a packet transmission protocol with a limited set of frame sizes. A frame contains one or more signals and the size of a signal is assumed to be less or equal to the size of the largest frame. Each node transmits and receives signals, where a signal has one producer and one or more consumers. Each signal has a specified size and deadline. We assume that the period time of generation of new signal values is greater than the deadline of the signal. The nodes may or may not have a global synchronised time base.

3.2 Problem formulation

For each node the following problem has to be solved. Given a finite set $S = \{s_1, s_2, \ldots, s_n\}$ of signals with size $sz(s_i) \in N^+$ and a deadline *d s ◎◎N◎◎i* ( ) . We define a frame *f* as a collection of signals from *S*. Each frame has an associated transmission time *c f ◎◎N◎◎j* ( ) and a size *sz f ◎N ◎◎j* ( ), defined by the used communication protocol. The problem is now to find a mapping of *S* into a set of frames , $F = \{f_1, f_2, \ldots, f_l\}$ such that each *s S i ◎◎*is included as in equ. (1)

$$\sum_{\forall s_i \in f_j} sz(s_i) \le sz(f_j) \qquad \ldots (1)$$

which minimises the bandwidth utilisation measure as in equ. (2)

$$U = \sum_{\forall f_k \in F} \frac{c(f_k)}{\min_{\forall s_i \in f_k} (d(s_i))} \quad \ldots (2)$$

Each frame has to be transmitted with a rate that fulfils the deadline requirement on the signal with the shortest deadline in the frame *fi*. The objective is to map the signals into frames such that the bandwidth requirement *U* is minimized, while making sure that frames are schedulable. This problem is *NP*-hard in the strong sense since it easily can be shown that it is a special case of the well known "bin packing" problem, which is a *NP*-hard combinatorial optimization problem [7]. The "bin packing" problem is obtained when all signals have the same deadline and when there is only one size of frames. Then our optimization problem becomes to pack the signals in as few frames as possible, which is exactly the "bin packing" problem. So if our problem is proven to belong to class *P* then should also the "bin packing" problem belongs to that class, which is a contradiction, unless *P* = *NP*.

3.3 An engineering approach: mapping signals to frames

The frame-packing problem is a NP-hard problem and hence we need to solve the problem by using heuristic techniques. To get a measure of the effectiveness of our algorithms, a theoretical lower bound for the utilization is derived for the signals. This theoretical lower bound is never higher than the real lower bound. The lower bound is calculated by assuming that each signal is transmitted in a frame with the lowest cost per bit and the deadline of the frame is the same as the deadline of the signal. A frame has a transmission time and a data size. We define the lowest theoretical overhead per bit by equ. (3)

$$MINOH = \min_{\forall f} (c(f)/sz(f)) \quad \ldots (3)$$

The minimal theoretical signal utilization, *SU*, for a signal *s* is calculated as in equ. (4) by

$$SU(s) = \frac{sz(s)}{d(s)} \times MINOH \quad \ldots (4)$$

The theoretical lower bound of the utilization for all signals is calculated as in equ. (5)

$$LB(S) = \sum_{\forall s \in S} SU(s) \quad \ldots (5)$$

Intuitively, this corresponds to packing each signal in a minimum overhead frame, together with other frames with the same deadline that completely fills up the frame. Our heuristic approach is to first sort the signals in increasing deadline order and then pack the signal into frames by a heuristic algorithm. We will consider two type cases of packing, the first packing algorithm (fixed frame size) considers only one size of the frames and exploits the first fit algorithm and the second algorithm (linear frame selection) uses heuristics for deciding which frame size to be used. A more detailed description of the algorithms can be found in [6].

3.4 Linear Frame Selection

The algorithm for fixed size frames assigns signals to frame until a signal does not fit into the frame, then a new frame is created and the signal is assigned to that frame. The algorithm starts off with a frame of the smallest frame size and assigns signals to that frame. When a signal *s* does not fit into the frame a selection is made; the cost (in bandwidth usage) for using a larger frame that fits all signals including *s* is compared with the cost of keeping the original frame and assigning *s* to a new frame with the smallest possible size. The alternative with the lowest cost is preferred. Moreover, when several frames have been created the algorithm first traverses the frames in order, trying to fit the signal into some unused space. If that is not successful the procedure described earlier is started. A nice property of both algorithms presented is that they are polynomial time algorithms which in practice mean that they are very fast to run even for large signal sets [17]

4. SIMULATION

To evaluate the quality of the proposed scheme analysis was made on four type-cases of signal sizes and deadline Distributions, both for a Controller Area Network (CAN) [10] based system and the slow and low cost Local Interconnection Network (LIN)[8]. CAN is a broadcast bus designed to operate at speeds up to 1 Mbps. Data is transmitted in frames containing between 0 and 8 bytes of data. A LIN installation usually runs at the speed of 5-20 Kbps/s and is intended to be used for control of internal lights, window drivers, selection switches, etc. in automotive systems. Data is transmitted in frames

containing 2, 4 or 8 bytes of data. By choosing two buses because they operate on different speeds and have different sets up of possible frame sizes. Further, a CAN based system is more likely to be used for sending larger signals in terms of number of bits since it is mostly used for sending control data, while the LIN based system is mostly used for replacing simple on/off logic. The sizes and

deadline distributions for each bus have been derived from discussions with our industrial partners [14].

To generate signal sets we have developed a test case generator that takes the following as input.

i. The theoretical lower bound bandwidth, which is used for regulating the amount of signals to be generated.
ii. The distribution of signal sizes (e.g., 70% of 1 bit signals, 20% of 2 bit signals and 10% of 3 bit signals)
iii. The distribution of deadlines (e.g., 20% of the signals has a deadline of 10, 25% of the signals have a deadline of 25 etc.)

4.1. CAN simulation

Signals were created with a distribution of the signal size as shown in Fig. 1 and each signal was given one out of nine different deadlines. Fig. 1 gives also the probability for assigning a specific deadline to a signal.

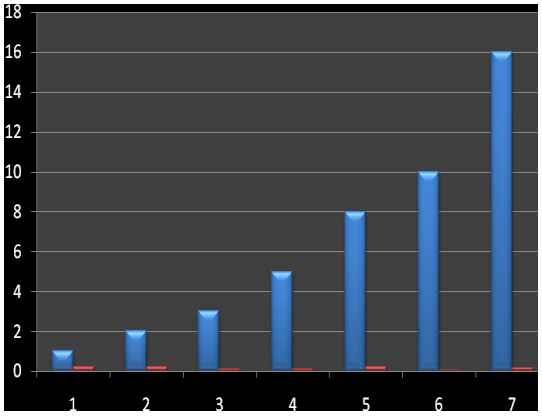
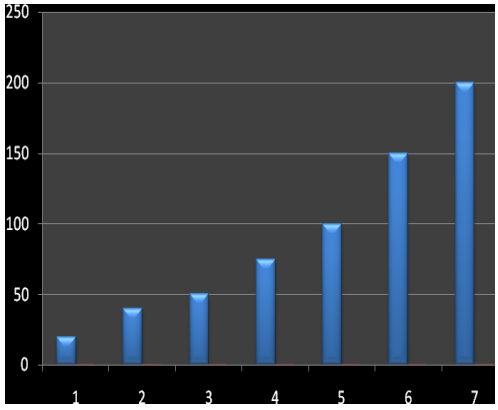


Fig. 1Probability Distribution of Signal sizes

Legends used in Fig. 1 are ---------: size, --------: Probability



.

Legends used in Fig. 2 are ---------: size, --------: Probability

Fig. 2 Distribution of Deadline distribution

The graph presented in Fig. 3 shows the bandwidth utilization of the frames as a function of generated signal sets with different loads. The graph was obtained by running 10000 generated signal sets for each load level. The graphs include the result from the algorithm and the fixed frame size algorithm. The fixed frame size algorithm was executed for 8 different frame sizes, however smaller CAN frames have been omitted as they result in much higher bandwidth utilization. The network was assumed to operate at 500 Kbps.

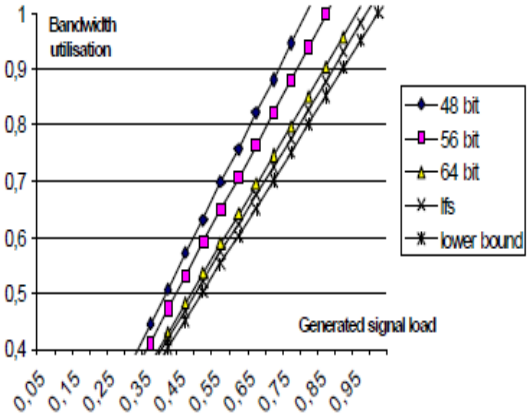


Fig 3. The performance at different load levels.

.4.2 LIN Simulation

Signals were created with a distribution of the signal size and each signal was given one out of seven different deadlines. Fig. 4 gives also the probability for assigning a specific deadline to a signal. The cost for the three different frame sizes was assumed to be 15, 20 and 25 respectively.

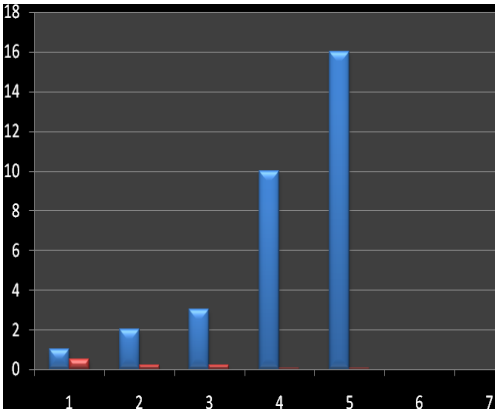


Fig 4. The Probability Distribution of Signal sizes

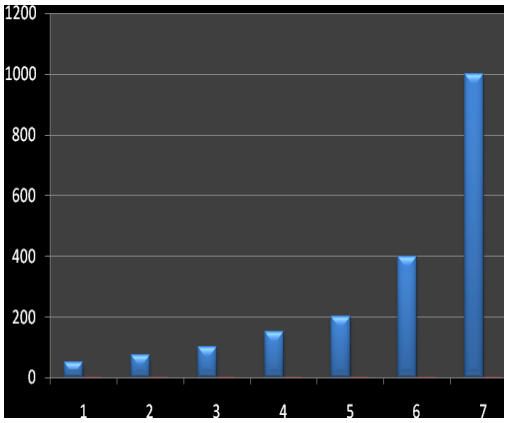


Fig. 5 Distribution of Deadline distribution

The graphs presented in Fig. 5 shows the bandwidth utilization of the frames as a function of generated signal sets with different loads. The graphs were obtained by running 10000 generated signal sets for each load level.

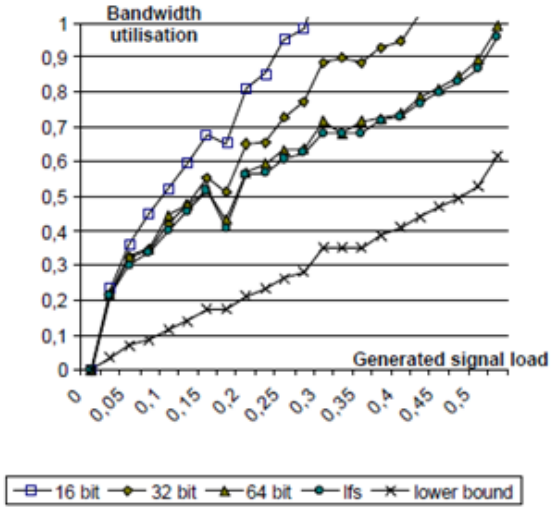


Fig. 6 The performance of the algorithms at different load levels generated

Legends used in Fig. 4 and Fig. 5 are ---------: size, --------: Probability

For small signal sets the 16 and 32 bit frames in the LIN simulation gives better performance than the 64 bit frame because the effect of not filling up the "last" frame is less significant. Further, compared to the CAN simulation, the LIN simulation also has a larger gap between the lower bound and the lfs algorithm since the price of not filling up the last frame is much higher (because the CAN-bus runs on a much higher speed), the CAN simulation includes significantly more signals and frames, and that only 3 frame sizes can be used in LIN.

5. CONCLUSION

In this paper, the frame-packing problem, is presented along with a formalisation of the problem, and showed that the problem is NP-hard, presented a heuristic solution, and demonstrated the heuristics effectiveness on signal sets that have been derived from real automotive applications. The results from this paper can be used for many different communication networks where several small signals have to share the space available in one frame. Further research includes looking into the issue of adjusting the period times of the frames in an efficient way. An interesting theoretical problem is to find out if it is possible to find an approximation algorithm, which can give a worst case upper bound on the waste of bandwidth for the algorithms presented in this paper. The CAN simulation includes many more signals, and hence more frames, than the LIN

tests and thus the price of not filling up the "last" frame is less significant.

(i) For small signal sets the 16 and 32 bit frames in the LIN simulation gives better performance than the 64 bit frame,because the effect of un-used space in the last frame is in average much higher.

(ii) It is quite easy to construct "pathological" cases where for example the 64 bit fixed frames behave much worse than the other algorithm.

(iii) Since all algorithms are so cheap to run one can always select the best result provided by any of the algorithms

6. REFERENCES

[1] Take F. Abdelzaher and Kang G. Shin. "Period-Based Load Partitioning and Assignment for Large Real-Time Applications", IEEE Transactions on Computers, 49, 2000.

[2] N.C. Audsley, A. Burns, M.F. Richardson, and A.J. Wellings. Hard Real- Time Scheduling, "The Deadline Monotonic Approach", In Proceedings of 8th IEEE Workshop on Real-Time Operating Systems and Software, 1991.

[3] Brenda S. Baker, "A new proof for the first-fit decreasing bin-packing algorithm", Journal of Algorithms, 6:49–70, 1985.

[4] Coffman, E.G., Jr., M.R. Garey, and D.S. Johnson, "Bin Packing with Divisible Item Sizes", Journal of Complexity, 3(4),406-428, December1987.

[5] Dionisio de Niz and Raj Rajkumar. Glue-code Generation, "Closing the Loophole in Model-based Development", In IEEE Real-time Systems and Application Simposium (RTAS) 2004 - MoDES, Toronto, CANADA, June 2004.

[6] Dionisio de Niz and Raj Rajkumar. Time Weaver, "A Software-Through- Models Framework for Embedded Real-Time Systems", In Proceedigs of ACM Language Compilers and Tools for Embedded Systems Symposium 2003, San Diego, CA, June 2003.

[7] D.S. Johnson. Near-Optimal Bin Packing Algorithms. PhD thesis, Massachusetts Institute of Technology, 1973.

[8] D.S. Johnson, A. Demers, D. Ullman, M.R. Garey, and R.L. Graham. "Worst-case performance bounds for simple one-dimensional packing algorithms", SIAM Journal of Computing, 3:299–325, 1974.

[9] Mark H. Klein, Thomas Ralya, Bill Pollak, and Ray Obenza." A Practitioners' Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems", Kluwer Academic Publishers, 1993.

[10] C. L. Liu and J. Layland, "Scheduling alghorithms for multiprogramming in a hard real-time environment", Journal of the ACM, Vol. 20, No. 1, 1973.

[11] Nir Naaman and Raphael Rom, " Packet Scheduling with Fragmentation", In Proceedings of the IEEE INFOCOM Symposium 2002, New York, NY, June 2002.

[12] Ken Tindell, Alan Burns, and Andy Wellings, "Allocating Hard Real- Time Tasks", An NP-Hard Problem Made Easy. Real-Time Systems, 4(2):145–65, 1992

[13]Tindell K., J. Clark, "Holistic Schedulability Analysis for Distributed Hard Real-time Systems". Technical Report YCS197, Real-Time Systems Research Group, Univ. of York, 1993.

[14] K. W. Tindell and A. Burns. "Guaranteed message latencies for distributed safety-critical hard real-time control networks",Technical Report YCS229, Dept. of Computer Science, University ofYork, June 1994.

[15] J Jonsson and J Vasell, "Evaluation and comparison of task allocation and scheduling methods for distributed real-time systems", In proceeding of Second IEEE International Conference on Engineering of Complex Computer Systems, 21-25 Oct. 1996.

[16] H. Jiandong and D. Ding-Zhu, "Resource management for continuous multimedia database applications"., In proceedings of RTSS'94. 1994.

[17]Hongfeng Wang; Yanjie Chen, "A Hybrid Genetic algorithm for 3D Bin Packing Problem" 2010 IEEE Fifth International conference, ISBN:978-1-4244-6437-1, 23-26 Sept. 2010

BIOGRAPHIES

1. Mr. M. Anand has completed his B.Tech. in the faculty of ECE and M.Tech. in the area of Applied Electronics. He is presently a research scholar in the faculty of ECE at St. Peters University, Avadi, Chennai. Mr. M.Anand shows interest in the areas of Embedded system design, Co-design of embedded systems, Scheduling etc

2. Dr. S. Ravi completed A.M.I.E. (Electronics Engineering) from Institution of Engineers, Calcutta in the Year 1991, M.Tech. (Communication systems) from Anna University, Chennai in the Year 1994 and Ph.D. from Anna University in the year 2004. He has got 17 Years of Teaching Experience and 3 Years of Industrial Experience. Presently, he is working as Professor and Head of the Department of Electronics Engineering in Dr.M.G.R. University, Chennai. He has so far published nearly thirty five papers in referred International Journals and successfully guided many Ph.D. scholars. He is a Life Member and Charted Engineer in Institute of Engineers (I) and Life Member of Indian society for Technical Education.

3. Mrs.V.Kavitha has completed his B.Tech. in the faculty of ECE and M.Tech. in the area of Applied Electronics. He is presently a research scholar in the faculty of ECE at Dr.M.G.R. University, Chennai. Her areas of interest are embedded system, Fault Diagnosis of VLSI Circuits etc

4. Dr.Uma rajaram completed B.E (ECE) from college of Engg., Guindy in the year 1977, M.E. (Communication System) from Anna University in the year 1990 and Ph.D. from Anna University of in the year 2010. She has got 28 years Industrial teaching experience and two years of experience. She is presently working as a Professor and Dean of E&T. she has published many papers in International Conference and journals. She is a life member of Indian society for technical education and fellow member in the institution of electronics and telecommunication engineers.