# A FRAME PACKING METHOD TO IMPROVE THE SCHEDULABILITY ON CAN AND CAN-FD

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

GÖKHAN URUL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

FEBRUARY 2015

Approval of the thesis:

**A FRAME PACKING METHOD TO IMPROVE THE SCHEDULABILITY ON CAN AND CAN-FD**

submitted by **GÖKHAN URUL** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Gönül Turhan Sayan _____
Head of Department, **Electrical and Electronics Engineering**

Assoc. Prof. Dr. Ece Güran Schmidt _____
Supervisor, **Electrical Electronics Engineering Dept., METU**

Assoc. Prof. Dr. Klaus Schmidt _____
Co-supervisor, **Mechatronics Engineering, Cankaya Univ.**

**Examining Committee Members:**

Assoc. Prof. Dr. Cuneyt Bazlamaçcı
Electrical Electronics Engineering Department, METU _____

Assoc. Prof. Dr. Ece Güran Schmidt
Electrical Electronics Engineering Department, METU _____

Assoc. Prof. Dr. Ilkay Ulusoy
Electrical Electronics Engineering Department, METU _____

Utku Karakaya, M.Sc
TOFAS ARGE _____

Vakkas Çelik, M.Sc
TUBITAK UZAY _____

**Date:** _____

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Last Name: GÖKHAN URUL

Signature :

# ABSTRACT

A FRAME PACKING METHOD TO IMPROVE THE SCHEDULABILITY ON CAN AND CAN-FD

Urul, Gökhan
M.S., Department of Electrical and Electronics Engineering
Supervisor : Assoc. Prof. Dr. Ece Güran Schmidt
Co-Supervisor : Assoc. Prof. Dr. Klaus Schmidt

February 2015, 60 pages

Controller Area Network (CAN) is the most widely used in-vehicle network. Today, vehicle applications can fill a CAN network's communication bandwidth to its limit. Hereby, the consumed bandwidth of an in-vehicle application depends on the efficiency of packing signal data into CAN message frames and on the suitability of the CAN message priority assignment such that all messages are schedulable. This thesis focuses on the problem of signal packing which is known to be and NP-hard problem. To this end, the thesis first investigates and implements the existing signal packing approaches. Based on this investigation, the thesis proposes a new signal packing heuristic that aims at minimizing the consumed bandwidth and at the same time obtain a schedulable message set. It is further shown that the developed method is not only suitable for CAN but also for CAN with Flexible Data Rate (CAN-FD) which is an extension of CAN with a higher data rate. The results of our extensive systematically conducted computational experiments show that our heuristic provides better results than existing techniques at a low run-time.

Keywords: Controller Area Network (CAN), CAN with Flexible Data Rate (CAN-FD), Schedulability analysis, Signal packing, Response time analysis, Optimization

# ÖZ

CAN VE CAN-FD VERİ YOLUNDA ZAMANLAMA İYİLEŞTİRMEK İÇİN BİR VERİ PAKETLEME YÖNTEMİ

Urul, Gökhan
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Ece Güran Schmidt
Ortak Tez Yöneticisi : Doç. Dr. Klaus Schmidt

Şubat 2015 , 60 sayfa

Denetleyici Alan Ağları (CAN) araç içi iletişim için kullanılan en yaygın haberleşme ağıdır. Günümüzde, taşıt uygulamaları CAN ağlarındaki band genişliği kullanımını üst sınırlarına çıkarmıştır. Bir taşıt uygulamasının kullandığı band genişliği sinyal verilerinin CAN mesajlarına paketlenmesindeki verime ve tüm mesajlara zamanlanabilir bir şekilde öncelik atanmasına bağlıdır. Bu tez, NP zor olarak bilinen sinyal paketleme problemi üzerine odaklanmaktadır. Öncelikle, mevcut sinyal paketleme yaklaşımları araştırılmış ve gerçeklenmiştir. Literatür araştırmasına dayanarak, zamanlanabilir bir mesaj kümesi oluşturularak band genişliği kullanımını azaltmayı amaçlayan yeni bir sezgisel paketleme yöntemi önerilmektedir. Çalışmalarımız, geliştirilen yöntemin sadece CAN ağları için değil CAN ağlarının daha yüksek veri hızına sahip olan genişletilmiş bir versiyonu CAN Esnek Veri Hızı (CAN-FD) için de uygulanabilir olduğunu göstermektedir. Kapsamlı ve sistematik bir şekilde yürüttüğümüz deneyler, önerdiğimiz sezgisel yöntemin mevcut tekniklere göre daha iyi sonuçları daha düşük

çalışma zamanı ile sağladığını göstermiştir.

Anahtar Kelimeler: Denetleyici Alan Ağı (CAN), Esnek Veri Hızı ile Denetleyici Alan Ağı (CAN-FD), Çizelgeleme analizi, Sinyal paketleme, Yansıma zamanı analizi, Optimizasyon

*To My Wife, Asmin*

# ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my supervisors Assoc. Prof. Dr. Ece Güran Schmidt and Assoc. Prof. Dr. Klaus Werner Schmidt for their continuous support, understanding and patience throughout my MSc. I am especially grateful for their confidence, time and the freedom they gave me to do this work. I would also like to express my gratitude to Utku Karakaya, Duygu Culum Karani and Onur Canbaz from the TOFAS R&D team for their contribution to my research.

I would like to thank my beloved wife, Asmin, for her love, kindness, support and sacrifice she has shown during the past two years it has taken me to finalize this thesis. Special thanks to my family. Words cannot express how grateful I am to my mother, father, mother-in-law and father-in-law for all of the encouragement and trust that you've given on my behalf. I would also like to thank to my brother, my business partner, Gökalp, who have given me extensive support and friendship throughout my thesis.

In addition, I thank to Mine Çetinkaya, who helped and encouraged on writing my thesis. I also thank ASELSAN and my ex-colleagues for support to conduct this thesis during my working life.

# TABLE OF CONTENTS

# LIST OF TABLES

TABLES

# LIST OF FIGURES

FIGURES

# LIST OF ALGORITHMS

ALGORITHMS

# LIST OF ABBREVIATIONS

| | |
|---|---|
| $ECU$ | Electronic Controller Unit |
| $CAN$ | Controller Area Network |
| $CAN-FD$ | Controller Area Network Flexible Data Rate |
| $BFD$ | Best Fit Decreasing |
| $FFD$ | First Fit Decreasing |
| $WCTT$ | Worst Case Transmission Time |
| $1SpF$ | One Signal Per Frame |
| $BBFd$ | Bandwidth Best Fit Decreasing |
| $BDFF$ | Bi-directional Frequency Fit |
| $IFPH$ | Improved Frame Packing Heuristic |
| $APBH$ | Advanced Period Based Heuristic |
| $CaFeS$ | CAN-FD Frame Selection Heuristic |
| $PBH$ | Period Based Heuristic |
| $OSI$ | Open Systems Interconnection |
| $FPP$ | Frame Packing Problem |
| $BPP$ | Bin Packing Problem |

# CHAPTER 1

# INTRODUCTION

Over the last few decades, the electronic revolution has been one of the biggest challenges of the automotive industry. For premium cars, "the cost of software and electronics can reach 35 to 40 percent of the cost of a car," as stated by Broy [6]. Controller Area Network (CAN) is the most widely used network in automotive applications to establish the network between electronic control units (ECUs) in vehicles [13]. Although CAN has been introduced more than 30 year ago, it is still widely used in series vehicles and there is still a large amount of ongoing research since complexity of today systems can fill a CAN network's communication bandwidth to its limit [11]. In order to extend limits while keeping most of the software and hardware – especially the physical layer – unchanged, a new protocol CAN-FD [11, 2, 1] has been proposed to increase bandwidth consumption. Two main extensions to CAN come up with CAN-FD, longer data fields and shortening the bit time during data transmission.

In-vehicle electronic systems contain ECUs which typically get its inputs from sensors (such as speed, acceleration, temperature, etc.) and that provide outputs to actuators (change gear, adapt velocity, etc.) to enforce actions determined. While an ECU is executing its local control and measurements, it needs to exchange data with other ECUs to achieve vehicle system wide tasks. This communication is performed on the CAN bus, where data (signals) is sent in message frames [17]. Since tasks in-vehicle operations are time-critical, signals should be exchanged with real-time constraints such as deadlines which are supported by CAN. To establish real-time constraints, in-vehicle signals have some important properties such as; bit length, period and deadline. CAN messages which contain multiple signals have the same

properties as the included signals and also have an additional important property: the priority that is used for arbitration and schedulability on the bus. To obtain a schedulable CAN network which ensures that each signal arrives at its destination before its deadline, packing signals into message frames efficiently and assigning suitable message priorities are essential.

The problem studied in this thesis is finding a mapping of all signals to size limited frames and assigning priorities to each frame such that the bandwidth utilization on CAN is minimized and the network becomes schedulable. That is, we assume that a set of signals to be sent over the network is given for each ECU. Each signal has a bit size, period and deadline which should be satisfied. Signals from the same ECU can be packed into CAN frames whose payload varies between 0 and 8 bytes. In addition, CAN frames have a payload-dependent overhead. Hereby, period and deadline of the frames depend on the packed signals. While packing, it is desired that signals are assigned to frames such that the overall bandwidth utilization on the CAN bus is minimized by eliminating overhead. This problem is known to be NP-hard [16] that the solution is thus to find efficient heuristics.

Prior to our study, there have been many studies in the literature that proposed frame packing approach for CAN. Sandstorm [16] states Bandwidth Best Fit decreasing (BBFd) method which is inspired from classic bin packing. Saket and Navet [15] propose Bi-directional Frequency Fit (BDFF) as a more effective strategy which takes into account the effect of the signal period. Pölzlbauer [14] states the Improved Frame Packing Heuristic (IFPH) that brings an additional optimality criterion to the proposed solution which tries to improve previous studies. Finally Bordoloi [4] proposes a framework (CaFeS) that is especially targeted for CAN-FD which claims to solve the packing problem in pseudo-polynomial time. Bordoloi is applying his approach to standard CAN with keeping maximum frame size at 8 bytes.

Difference between CAN and CAN-FD in the scope of frame packing problem is Worst Case Transmission Time (WCTT). When WCTT is properly adapted to the existing approaches, all the packing solutions will be available for both CAN and CAN-FD. Hence, the development of this thesis focuses on CAN and the findings are then also applied to CAN-FD with small modification on WCTT equation.

Except for BBFd, the existing approaches focus on the period property of the signals such that the ideal situation is obtained when having fully utilized frames with signals of the same period. Our approach also considers the signal period but with a different approach. We derive a heuristic called Advanced Period Based Heuristic (APBH). Our method starts with packing all signals with the same periods using a bin packing heuristic. As a result, we already obtain an efficient message set after this step considering that the generation period of signals in the same message are identical. Nevertheless, it is possible that some of the messages are not fully filled. Hence, the second step of our method disassembles unnecessarily small frames and distributes the respective signals to other frames. To compare with the existing studies, all algorithms in the literature are implemented in our software framework. Experiments are conducted with all approaches systematically using randomly generated signal sets with different properties. Our new algorithm APBH leads to packing solutions with the lowest bandwidth utilization on schedulable frame sets for all types of input signal sets.

In summary, the contributions of the thesis are as follows:

1. Implementation of the signal packing methods BBFd, BDFF, IFPH, CAFeS in [16, 15, 14, 4] in a common software tool. All heuristics are implemented both for CAN and CAN-FD. In the existing literature, implementations for CAN-FD are not available.

2. Development and implementation of our new signal packing heuristic APBH for CAN and CAN-FD,

3. Extensive comparison of the existing methods and our method for both CAN and CAN-FD using randomly generated signal sets with different properties. The comparison shows that our method gives better results in all cases.

The remainder of this thesis is organized as follows. Chapter 2 gives a description of background information on CAN and CAN-FD protocols. Chapter 3 describes the signal packing problem in detail, specifically, properties of the packing problem, existing methods in the literature and the proposed new approach. The experimental setup, properties of the signals sets used as input in the experiments, result of ex-

periments and a detailed discussion are given in Chapter 4. Finally, in Chapter 5, conclusions are drawn from our study.

# CHAPTER 2

# BACKGROUND

## 2.1 CAN Controller Area Network

CAN is a serial communications bus that is initially developed for the automotive industry to replace the multi-wire cabling. It is an International Standardization Organization defined term that is structurally composed of a two-wire bus. The characteristic properties of the CAN is that it has high immunity to electrical interference and it enables self-identification and correction of data errors. These specific properties have made its widespread usage in other industries, such as building automation, medical, and manufacturing. (reference)

Furthermore, the CAN communication bus complies with the ISO-11898: 1993 [17] CAN communication protocol that specifies the data link layer and physical signaling of the CAN. This protocol describes how information is transmitted among devices and also complies with the layer-based defined Open Systems Interconnection (OSI) model. The physical layer of the model defines how communication is carried out among the devices that are a part of this physical medium. In figure 2.1 given below, the ISO 11898 architecture depicts the data-link and physical layers as the lowest two layers among the seven-layered structure. [9]

Reflecting back to our research, in our study, we don't investigate the specific properties of physical and data-link layer of CAN that is described above since our study involves specifically the application layer. However, for instance, in frame length computations, bit stuffing that is carried out on data-link layer are also in our context. CAN frames that are used in the application layer are basically composed of 8 bytes

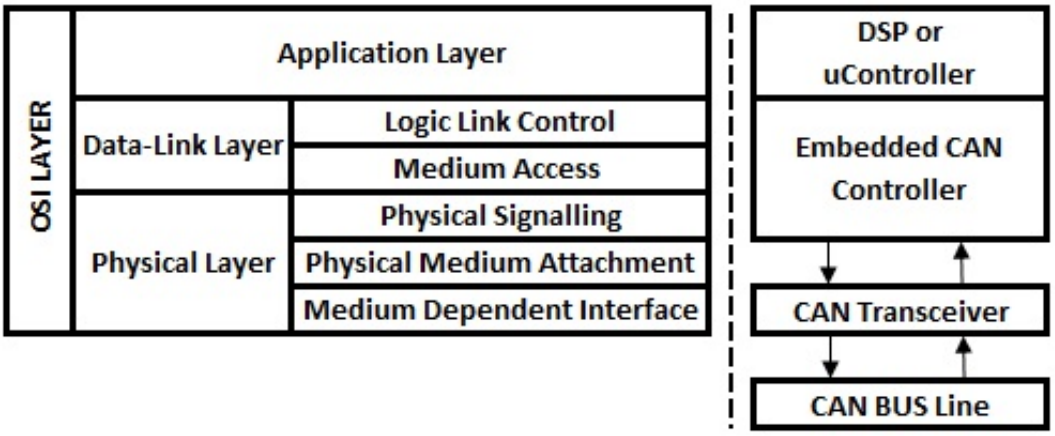



Figure 2.1: The Layered ISO 11898 Standard Architecture

of data, 1 bit RTR, and the arbitration ID. The details of these CAN frames existing in the data link layer, such as overhead and checksum, are given under the next section that is describing the CAN terminology and figure 2.2.

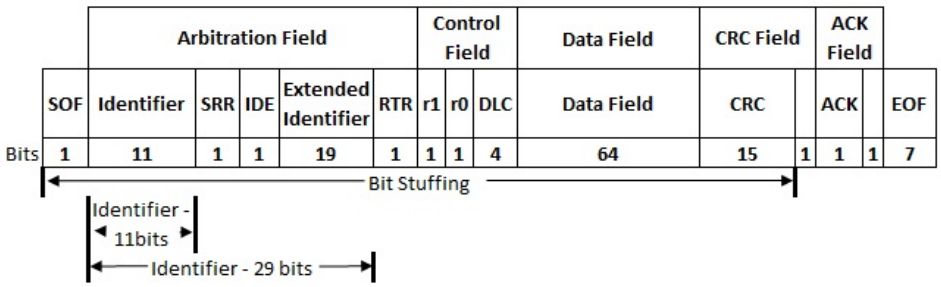



Figure 2.2: std CAN Frame Format

| | |
|---|---|
| **CAN Frame** | CAN Frame is the entire CAN message or frame including arbitration ID, data bytes, acknowledge bit, etc. |
| **SOF** | Start Of Frame (SOF) indicates the beginning of a message with a dominant (logic 0) bit. |
| **Arbitration ID** | identifies the message and indicates the message's priority. Frames come in two formats – standard, which uses an 11-bit arbitration ID, and extended, which uses a 29-bit arbitration ID. |
| **IDE** | Identifier Extension Bit (IDE) allows differentiation between standard and extended frames. |
| **RTR** | Remote Transmission Request (RTR) Bit serves to differentiate a remote frame from a data frame. A dominant (logic 0) RTR bit indicates a data frame. A recessive (logic 1) RTR bit indicates a remote frame. |
| **DLC** | Data Length Code (DLC) indicates the number of bytes the data field contains. " |
| **Data Field** | contains 0 to 8 bytes of data. |
| **CRC** | Cyclic Redundancy Check (CRC) contains 15-bit cyclic redundancy check code and a recessive delimiter bit. The CRC field is used for error detection. |
| **ACK** | ACKnowledgement slot, any CAN controller that correctly receives the message sends an ACK bit at the end of the message. |
| **CAN Signal** | It is an individual piece of data contained within the CAN frame data field. |

As CAN is a serial data bus that supports priority based message arbitration and non-preemptive message transmission, in this section, we will highlight the priority based arbitration and non-preemptive message transmission properties of CAN, since these properties are one of the most beneficial features that make the CAN very suitable as a real time prioritized communications system.

In our research, we focus on the scheduling of the transmitted messages in a CAN bus and the priority-based arbitration feature plays a crucial role in the scheduling.

As priority-based arbitration favors the dominant coding over recessive coding, in the early 1990s, there was a common misunderstanding about the CAN indicating that it sustains highest priority messaging with low latency, whereas it cannot guarantee the deadline of the less urgent, lower priority messages. [10]

The closely related feature of CAN, which are priority based arbitration and the pre-emptive scheduling are studied further in the literature to overcome the biased approach to CAN in the early 1990s. For instance, the adaptation and application of fixed priority pre-emptive scheduling of single processor systems to the scheduling of the messages on CAN were investigated in the research of scholars Tindell and Burns [18] and Tindell et al. [19, 21]. This message scheduling analysis made itself useful as a way to calculate the worst-case scenario for all CAN messages response times. In the light of this analysis, one can design CAN based systems that have correct timing and that ensure the deadlines of all messages and signals transmitted. [10]

When bus utilization of CAN is considered before Tindell's work, it was common to have around 30 or 40 % low levels of bus utilization in automotive applications. Hence, in order to have assurance that CAN messages would meet the deadlines, large scale testing was essential. Following the introduction of the systematic approach that is based on schedulability analysis which is first introduced by Tindell, approximately 80 % utilization was achievable for CAN bus utilization, as long as ensuring that the deadline would be met. [10]

## 2.2 CAN Controller Area Network Flexible Data Rate

Many CAN buses have reached 50% - 95% bus load level. [11] Standard CAN messages contain $\geqslant$ 50% overhead: standard CAN requires $\approx$ 129 bits/message for 64 bits of data and extended CAN needs $\approx$ 154 bits/message for 64 bits of data. At a given bus load, only $\approx$ 40-50% of the bandwidth is used to exchange useful data. At the same time, CAN comes with low data rates of $\leqslant$ 1Mbit/sec. The acceptance and introduction of serial communication to more and more applications has led to increasing demand for bandwidth in CAN communication and caused system developers to look

for alternative communication options in certain applications.

These applications can be realized more comfortably with the new protocol CAN-FD that allows data rates higher than 1 MBit/s and payloads beyond 8 bytes per frame. Basic idea of CAN-FD is to increase the bandwidth of a CAN network while keeping most of the software and hardware. To achieve the idea, CAN-FD has two main principles; first one is using frames with more than 8 data bytes and second one is to switch to a different bit rate after the arbitration. CAN-FD is a serial communication protocol compatible with ISO 11898-1. CAN-FD shares the physical layer, with the CAN protocol as defined in the BOSCH CAN Specification 2.0. The frame format however, is different. [11] Frame format of CAN-FD and terminology is defined in figure 2.3 as;

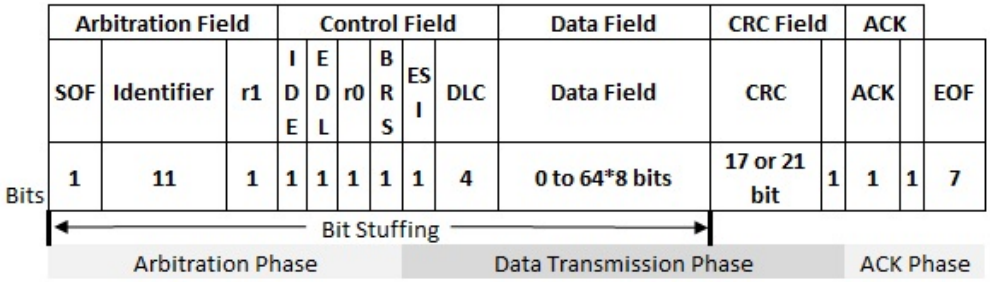



Figure 2.3: CAN-FD Frame Format

| | |
|---|---|
| **SOF** | Start of frame (bit is always of dominant state). |
| **ID** | Identifier (frame priority and content indication). |
| **rtr** | Remote transmission request (dominant, if data frame). |
| **IDE** | ID extension (dominant for base frame format). |
| **EDL** | Extended data length (recessive, if data field is longer than 8 byte). |
| **r0/r1** | Reserved bits. |
| **BRS** | Bit rate switch (recessive, if switched to alternate bit-rate). |
| **ESI** | Error state indicator (recessive, if transmitting node is in error passive state). |
| **DLC** | Data length code (indicates the length of the following data field). |
| **CRC** | Cyclic redundancy check (15-bit, 17-bit, or 21-bit). |
| **D** | Delimiter of CRC/ACK field (bit is always of recessive state). |
| **ACK** | Acknowledgment slot (correctly receiving node sends a dominant bit). |
| **EOF** | End of frame (all bits are always of recessive state). |
| **IFS** | Inter-frame space (the first two bits are always of recessive state). |
| **IDLE** | Bus is in recessive state. |

To summarize, the main differences of CAN-FD over CAN are as follows:

- CAN-FD supports dual bit rates within a message frame; Arbitration-Phase – same bit rates as standard CAN, Data-Phase – sub-multiple of controller clock rate. Most importantly, the data-phase is intended to support bit rates $\geqslant$ than Arbitration Phase bit rate.
- CAN-FD supports larger data lengths than "standard" CAN (up to 64 bytes/message). Differences from CAN are limited to CAN-FD controller hardware, whereas the system cost is similar to standard CAN: Controller, crystal, transceiver and possibly wiring.
- The three bits FDF (FD Format), BRS (Bit Rate Switch), and ESI (Error Status

Indicator) are introduced for CAN-FD.

- All of the advantages of CAN are available in CAN-FD: message prioritization, guaranteed latency times, flexible configuration, multi-master, multi-cast capability, error detection and signaling, automatic retransmission on error.
- CAN-FD is a superset of CAN: Maintains CAN arbitration scheme, maintains ACK scheme, has mode that conforms with CAN 2.0 and ISO11898-1. CAN-FD adds, higher data bit rates, larger data fields (up to 64 bytes), larger CRC polynomials to handle larger data fields.

## 2.3 Frame Length Computation of CAN-FD and CAN

In this section, we present analysis that bounds the worst-case transmission time of a given CAN message. The worst-case transmission time of a given message $m$ is defined as the longest time between putting the CAN frame $m$ on the bus and the latest time that the message arrives at the destination stations. On the other hand, the best-case transmission time on CAN occurs when no stuff bits are inserted during frame transmission. Stuff bits increase the maximum transmission time of CAN messages. Hence, the worst-case transmission time includes stuff bit scenario. Since CAN operates a fixed priority scheduling algorithm and is not fully pre-emptive, a high priority message cannot interrupt a message that is already transmitting.

The term $s_m$ denotes the bounded size of a CAN frame $m$ in bytes where $s_m$ = 0,1,2,3,4,5,6,7,8. The term $\tau_{bits}$ is the bit time of the bus. The term $C_m$ represents the longest time taken to physically send message $m$ on the bus. Maximum transmission time $C_m$ of standard CAN frame is given [10] by:

$$C_m = g_{CAN} + 8s_m + 13 + \left\lfloor \frac{(g_{CAN} + 8s_m - 1)}{4} \right\rfloor \tau_{bits} \tag{2.1}$$

To obtain a standardized formula, we can write $C_m = O_{CAN} + f_{CAN}(s_m)$:

$$O_{CAN} = (g_{CAN} + 13 + \left\lfloor \frac{(g_{CAN} - 1)}{4} \right\rfloor)\tau_{bits} \tag{2.2}$$

$$f_{CAN}(s_m) = 10s_m\tau_{bits} \tag{2.3}$$

where $g_{CAN}$ is 34 for the standard format (11-bit identifiers) or 54 for the extended format (29-bit identifiers).

To express transmission time of a CAN-FD frame, we denote the term $\tau_{Arbits}$ as the bit time of the bus in the arbitration phase and the term $\tau_{Dabits}$ is the bit time of the bus in data phase. The term $s_m$ denotes the bounded size of a CAN-FD message m in bytes where $s_m$ = 0,1,2,3,4,5,6,7,8,12,16,20,24,32,48,64. Maximum transmission time $C_m$ of a CAN-FD frame is given by

$$C_m = \Big(g_{CANFD} + 12 + \Big\lfloor \frac{g_{CANFD}}{4} \Big\rfloor\Big)\tau_{Arbits} + \Big(28 + 5\Big\lceil \frac{s_m - 16}{64} \Big\rceil + 10 s_m\Big)\tau_{Dabits} \tag{2.4}$$

To obtain a standardized formula, we can write $C_m = O_{CAN} + f_{CAN}(s_m)$;

$$O_{CAN} = \Big(g_{CANFD} + 12 + \Big\lceil \frac{g_{CANFD}}{4} \Big\rceil\Big)\tau_{Arbits} + 28\tau_{Dabits} \tag{2.5}$$

$$f_{CANFD}(s_m) = \Big(5 \Big\lceil \frac{s_m - 16}{64} \Big\rceil + 10 s_m\Big)\tau_{Dabits} \tag{2.6}$$

where $g_{CANFD}$ is 17 for the standard format (11-bit identifiers) or 36 for the extended format (29-bit identifiers). In CAN-FD frames, CRC calculation has additional properties. For payloads up to 16 bytes, the CRC field is 17 bits. For payloads larger than 16 bytes, the CRC field is 21 bits. To obtain the stuff bit size change, in the formulas ceiling function $\lceil \frac{a}{b} \rceil$ is used.

CAN-FD frame transmission could be more efficient than standard CAN frames since the bit rate of the data phase of a CAN-FD frame could be higher and the data size of CAN-FD could reach up to 64 bytes. However, when arbitration and data bit rate of a CAN-FD frame are the same and carrying 8 bytes like a standard CAN frame, the transmission time of a standard CAN frame will be lower than the CAN-FD one. To obtain the relation between bit rates and data size, the following expression is used. $P$ denotes the efficiency of CAN-FD frame transmission time over standard CAN frame while sending 8 bytes of data.

$$P = \frac{CANFD\ C_m}{CAN\ C_m} \tag{2.7}$$

# CHAPTER 3

# FRAME PACKING

## 3.1 Problem Formulation

Basically, in a single CAN frame, there are arbitration and data bits, in which the arbitration bits can be considered as the overhead component of the frame. Even though the data size of the CAN frame may change, the arbitration size will not be affected. In other words, overhead to data ratio grows larger as the data size in the frame gets smaller. The most effective way to use a CAN line would be packing different, small-sized signals together and sending the bundle as a single frame so as to minimize the bandwidth requirement, while ensuring that the frames are schedulable.

Although, we have highlighted that frame packing is an efficient method to decrease the bandwidth utilization, assigning different sized signals to frames becomes a complex task since the signals are arriving asynchronously and their periods and transmission deadline requirements may differ. [18, 20] Briefly, in a CAN-based system, the data is transmitted in frame sizes which may change between 0 to 8 bytes of data and the signals may vary between 1 bit and 64 bits. Moreover, each signal may have different period and deadline. These differences in period, deadline, and data size bring complexity to the priority allocation of the signals in a CAN bus, in order to make the CAN bus schedulable. For example, packing these varying propertied signals in a frame requires assigning a new deadline, period, and priority to the new frame. This new frame's period should be assigned according to the signal that has the smallest period. As another frame property, the deadline assignment needs to be handled with care as is discussed in Section 3.2.

In principle, the frame packing problem is a more complicated case of the well-known, NP-hard [16] "bin packing" problem, which assumes that all the signals have the same deadline and there is a only single frame size. The basic setting of the frame packing problem can be summarized as given below:

- The set of frames is denoted as $F = \{f_1, f_2, \ldots, f_k\}$ for a given set of signals $S = \{s_1, s_2, \ldots, s_n\}$
- None of the signals transmitted miss their deadline while utilizing the smallest amount of bandwidth as possible
- In each ECU, resulting set of frames are schedulable
- Each signal is associated to the ECU where it is produced
- Each signal $s_i$ has 4 components ($N_i$, $t_i$, $c_i$, $d_i$)

  $N_i$ the ECU containing the signal

  $t_i$ the generation period of the signal on that ECU

  $c_i$ the size of of the signal in bits

  $d_i$ the deadline relative to the generation time of the signal $s_i$

In the context of our study, the deadline of a signal is equal to its period ($d_i = t_i$) as in state-of-the-art studies. [15, 16]

### 3.2 Properties of the Problem

In addition to the standard bin packing problem (BPP), the frame packing problem (FPP) has specific properties: in the FPP context, signal schedulability, message priority and deadline requirements need to be considered apart from simple packing. In this section, we focus on the specific FPP properties. Firstly, we shall start with one of the most important criteria in FPP, which is the deadline of a frame.

### 3.2.1 Deadline of A Frame After The Addition of A Signal

In his study, Marquoes [12] determines the effect of each additional signal on the frame deadline. In a frame that is composed of several signals, the transmission period of the frame is the smallest generation period of the signals, while the deadline of the frame is not equivalent to the smallest deadline of these signals. To illustrate this fact, consider two signals, denoted as $s_1$ and $s_2$, having respective periods of $T_1$=10 and $T_2$=14, and deadlines of $D_1$=10 and $D_2$=14. As seen in Fig. 3.1, the signal $s_2$, which is generated at time 14 could be transmitted at time 20 and $s_2$ generated at time 42 is sent at time 50. Due to the timing constraint of $s_2$, it is seen that the deadline of the frame must be equal to 8.

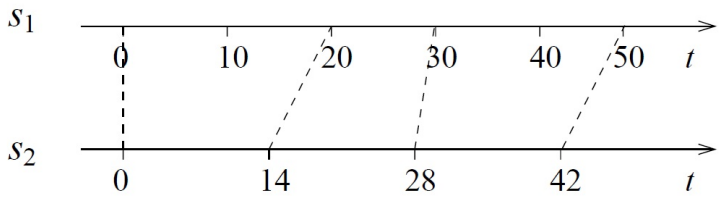



Figure 3.1: Two signals with production periods equal to 10 and 14. The dotted line indicates when the signal with period 14 is actually transmitted.[15]

Assume that a frame $f_k$ already contains signals $s_1, s_2, s_3, \ldots, s_n$. $s_m in$ is the the signal with smallest period. Then, the period of $f_k$ is $T_k^* = T_{min}$. The deadline of $f_k$ is $D_k^* = \min\{D_j - \text{Offset}(T_{min}, T_j)\}$ where Offset$(a, b)$ returns largest possible duration between the generation time of a signal with period $b \geqslant a$ and the transmission of the frame with period $a$ that contains the signal. In our context, offset formulation is as following [15];

$$\text{Offset}(a, b) = \left(\frac{a}{\gcd(a, b)} - 1\right) \cdot \gcd(a, b) = a - \text{gdc}(a, b). \tag{3.1}$$

$\gcd(a, b)$ denotes the greatest common divisor of $a$ and $b$. For the example case, the deadline should be set to 6.

### 3.2.2 Fully Utilized Frames and Period

Another property in the FPP context is that signals that are packed to form a new frame have effect on the frame period and bandwidth consumption. In order to decrease bandwidth utilization, the ideal situation would be having similar or same periods for all signals inside a frame and having fully utilized frames. However, the ideal case cannot be achieved most of the time, since the period and data size of the messages vary depending on each application. For a frame that is composed of signals with different periods, the frame period must be equivalent to the shortest period. Hence, signals with larger periods than the frame period may be transmitted more frequently than it is required. Consequently, this frequent transmission increases the bandwidth consumption. Besides, to reduce bandwidth consumption, frame utilization should be improved, and thus, the overhead ratio should be decreased. This trade-off in the FPP heuristic context is inquired further in the literature. Moreover, the state-of-the-art FPP heuristics will be described in detail in Section 3.3 .[16, 15, 14]

### 3.2.3 Bandwidth Utilization Computation

In a given signal set, periodic signals have properties of deadline, period and size. The transmission time of the messages fills the time taken in the bus. Bus utilization is an important property that has a big impact on schedulability. In our approach, CAN and CAN-FD are considered. Therefore a generalized formula to obtain bandwidth utilization is needed. Assume a finite signal set $S = \{ s_1, s_2, .., s_n \}$ with sizes $c_{s_i} \in \mathbb{N}$ and periods $t_{s_i} \in \mathbb{N}$. We define a frame $f$ with signals from $S$. Each frame has a size $C_{f_i} \in \mathbb{N}$, period $T_{f_j} \in \mathbb{N}$ and transmission time $C_{f_j} \cdot \tau_{\text{bit}}$. $F$ is the set of frames $F = \{f_1, f_2, .., f_n\}$. Then, the bandwidth utilization $U(F)$ of the frame set $F$ can be obtained by;

$$U(F) = \sum_{f_k \epsilon F} \frac{C_{f_k} \cdot \tau_{\text{bit}}}{T_{f_k}} \tag{3.2}$$

Recall that the transmission times $C_{f_i}$ are computed according to (2.1) and (2.4).

### 3.2.4 Frame Utilization Computation

As mentioned in Section 3.2.2, frame utilization is another important criteria for the efficient usage of CAN. Although putting a signal with higher period in a frame with lower period increases bus utilization, frames that fully use the available payload with signals of similar periods decrease the bus utilization. We will measure the Signal/Frame utilization of the proposed frame packing algorithms with the formula;

$$U_f(F) = \frac{\sum_{f_k \epsilon F} \sum_{s_m \epsilon f_k} c_{s_m}}{\sum_{f_k \epsilon F} C_{f_k}} \tag{3.3}$$

## 3.3 Frame Packing Algorithm

The motivation for proposing a new algorithm for the frame packing problem comes from the performance analysis of previous approaches to the problem. The frame packing problem is tackled with the following state-of-the-art frame packing heuristics;

1. One signal per frame (1SpF)
2. Bandwidth-Best-Fit decreasing (BBFd) [12]
3. Bi-directional Frequency Fit (BDFF) [15]
4. Improved Frame Packing Heuristic (IFPH) [14]
5. CAN-FD Frame Selection Heuristic (CaFeS) [4]

### 3.3.1 Bandwidth-Best-Fit decreasing (BBFd)

Marquoes [12] proposed in his study that Bandwidth-Best-Fit decreasing (BBFd) is more effective than the priorly stated "one signal per frame", First-Fit Decreasing (FFD), Best-Fit Decreasing (BFD) strategies. BBFd is also based on the "bin-packing" algorithm [7]. In the BBFd context, firstly the signals are sorted in decreasing order of bandwidth consumption as stated in Algorithm 1. Afterwards, starting

from the beginning of the sorted list, signals are inserted into the available frame with the minimum bandwidth comsumption. However, this strategy results in not fully packed frames and leads to the accumulation of signals with different periods. Because of these consequences, this strategy is not always an effective one as is shown in our experimental evaluation in Chapter 4.

```
1  Sort signals in decreasing order of bandwidth consumption
2  while no signal left do
3      Take signal
4      if at least one frame accepting the signal then
5          Find the frame with minimum bandwidth
6          Insert signal to this frame
7          Change frame timing
8      else
9          New frame with new timing
10         Insert signal to this frame
11         Change frame timing
12     end
13 end
14 Make feasibility test (Audsley algorithm)
15 if configuration is not feasible then
16     find the firt frame which violates its deadline. This frame is f
17     Take out the signal with smallest deadline, and put it in a new frame.
18     Update timings of f
19     go to line 14
20 else
21     Packing is completed!
22 end
```

**Algorithm 1:** Bandwidth-Best-Fit decreasing (BBFd) heuristic.

### 3.3.2 Bi-directional Frequency Fit (BDFF)

Navet [15] has proved that Bi-directional Frequency Fit (BDFF) is a more effective strategy than previously stated solutions, namely BBFd and 1SpF, when bandwidth consumption and schedulability are considered. The novelty in the BDFF heuristics lies in the grouping of same period signals together so as to reduce the bandwidth consumption as stated in Algorithm 2. In this heuristic, two frame sets are constructed as Front_set and Back_set. The approach basically tries to achieve that larger period signals are not packed in smaller period frames.

**input** : $S$ is the bi-directional list of signals
**output**: Construct $F = \{\}$ do $F = F_{front} \bigcup F_{back} \bigcup F$

```
1  Sort signals in increasing order of period in a bi-directional list S
2  Construct F_front = F_back = { } ; b_Front = true
3  if S is empty then
4      Go to step 31
5  else
6      if b_Front == true then
7          Remove a signal s_j from front of S, construct a new frame f_new in
           F_front, insert s_j in f_new and add f_new to F_front
8          Find the frame f in F_front which minimizes the bandwidth
9          Compare the frame f to the one obtained by inserting s_j in a new frame
10         if no frame exists then
11             b_Front = false
12             Go to step 3
13         else
14             Remove signal s_j from S
15             Put the signal s_j in frame f
16             Go to step 7
17         end
```

```
18 else
19     Remove a signal s_j from back of S, construct new frame f_new in F_back,
       insert s_j in f_new and add f_new to F_back;
20     Find the frame f in Fback which minimizes the bandwidth;
21     Compare the frame f to the one obtained by inserting s_j in a new frame;
22     if no frame exists then
23         b_Front = true;
24         Go to step 3;
25     else
26         Remove signal from S;
27         Put the signal f_j in frame f;
28         Go to step 19;
29     end
30 end
31 Feasibility test (Audsley algorithm);
32 if F is feasible then
33     Packing is completed!;
34 else
35     Find the frame in F containing at least 2 signals and which has the least
       difference between the worst case response time and the deadline at the
       lowest priority which has not been assigned.;
36     Remove the signal and make a new frame f containing only the signal ;
37     Add the new frame to F;
38     Go to step 31;
39 end
```

**Algorithm 2:** The Bi-directional frequency fit(BDFF) heuristic.

### 3.3.3 Improved Frame Packing Heuristic (IFPH)

On the other hand, Improved Frame Packing Heuristic( IFPH) Pölzlbauer [14] proposes an additional optimality criterion for decreasing bandwidth demand of each packing step as stated in Algorithm 3. The approach does not guarantee optimal packing but in the study (IFPH), Pölzlbauer obtains results that outperform state-of-the-art approaches which are 1SpF and BBFd. Pölzlbauer study does not contain any comparison with BDFF, which also outperforms BBFd that is stated in [15].

```
   input : S is list of signals
   output: F is list of packed frames
 1 Sort signals in S in increasing order of period
 2 Construct a new frame f
 3 foreach signal s in S do
 4     if left payload of f is larger than size of s then
 5         Apply optimality criterion [14]
 6         if Creating new frame is beneficial then
 7             Create new frame f and add s to f
 8         else
 9             Add s to f
10         end
11     else
12         Create new frame f and add signal s to f
13     end
14 end
```

**Algorithm 3:** The Improved frame packing heuristic.

### 3.3.4 CAN-FD Frame Selection Heuristic (CaFeS)

CAN-FD Frame Selection(CaFeS) [4] method is as well an optimization heuristic that intends to decrease bandwidth waste of the signal communication. The main idea of CaFeS is a dynamic programming algorithm to reduce the bandwidth waste problem with recursive equations. Bordoloi claims that CaFeS algorithm provides much better

results than Pölzlbauer's Improved Frame Packing Heuristic (IFPH) [14]. Since the claimed results in [4] could not be reproduced by our implementation, we refrain from giving a detailed explanation of this heuristic.

## 3.4 Proposed Frame Packing Algorithm

Referring to the previous discussion, BBFd basically approaches frame packing as a bin packing problem with the bandwidth issue. In the first step of BBFd, all signals are sorted in decreasing order of bandwidth consumption. After sorting, signals are basically packed according to best fit decreasing algorithm. Subsequent to BBFd, BDFF emphasizes periods of signals highlighting that same frequencies should be grouped together. This idea gives better results than previous state of art BBFd. IFPH uses an additional optimality criterion to solve the frame packing problem. That is, the superior approaches BDFF and IFPH already indicate that signals with the same period should be grouped together. However, they do not directly group the signals in same period groups. Differently, the first step of our approach is that signals should be grouped in same period signal sets. Afterwards, a specific optimality criterion will be used to increase the efficiency of the period-based packing.

### 3.4.1 General Observations and Motivation

Efficient use of bandwidth and schedulability are the main purpose of this study. Frame packing can minimize the overhead on the bus and decrease bandwidth utilization. Overhead data can be eliminated ideally by fully utilized frames with maximum frame size, and also packing signals with the same period together. Since data size as well as the period of signals vary in practical systems, we propose a heuristic for frame packing according to varying data size and periods.

If signals with varying periods are packed into a frame, the frame must have the lowest period. Therefore some of the messages may be sent more frequently than needed which results in waste on bandwidth. Besides, the more signals are packed into a frame, the less frames and consequently less overhead may be produced which results an increase on bandwidth utilization. For illustration, we consider two signals

$s_1$ and $s_2$ with periods $t_{s_1} = 10$ ms, $t_{s_2} = 100$ ms and sizes $c_{s_1} = 4$ bytes, $c_{s_2} = 4$ bytes. When $s_1$ and $s_2$ are packed together in a frame $f$, period of the new frame will be the lowest period of the signals $t_f = 10$ ms and size $c_f = 8$ bytes. If $s1$ and $s2$ were packed separately as 1SpF (1) in frames $f_{s1}$ and $f_{s2}$, there would be overhead $11 \cdot OH = 1 \cdot OH_{f_{s1}} + 10 \cdot OH_{f_{s2}}$. After the packing solution frame $f$ consists of two signals and period of $f$ is 10 ms. Then, the overhead is $10 \cdot OH$ which is decreases the bandwidth waste in terms of overhead. However, since $s_2$ is transmitted 10 times more than necessary in frame $f$, the bandwidth utilization is much higher when packing $s_1$ and $s_2$ together.

Our approach is composed of three distinct parts; The first part packs only signals with the same period into frames. The second part deals with disassembling frames and distributing signals to other available frames according to an optimality criterion. The final part consists of schedulability test of the found solution.

In the first part, we group signals with the same period in separate signal sets. Then, we use Best Fit bin packing to pack signals of each group in frames with a high signal utilization [7].

Secondly, although we have obtained frames with same period signals, some of the frames might only have afew signals which may create unnecessary overhead. Signals in higher period frames might be distributed to other frames which have available empty space. But a criterion is needed to make a decision which frames need to be disassembled and which signals could be distributed.

For illustration, we consider three frames $fn, n = 1, 2, 3$ with the following properties.

| | |
|---|---|
| $T_{fn}$ | message period |
| $pay_{fn}$ | payload of frame |
| $OH_{fn}$ | overhead of frame |
| $C_{fn}$ | size of message |

We assume that $t_{f1} < t_{f2} < t_{f3}$ and all signals have the same periods as the respective frames. $f_3$ consists of the signals $s_1$ and $s_2$, $c_{s1} = 1\,\mathrm{B}$ and $c_{s2} = 1\,\mathrm{B}$. Overhead of the frames is $OH_f n = 4\,\mathrm{B}$, $n = 1, 2, 3$. $f_1$ and $f_2$ have empty spaces where $s_1$ and $s_2$ could fit. We perform the following evaluation of the bandwidth utilization. On the

left hand side, we assume 3 frames, whereas on the right hand side we assume that $s1$ and $s2$ are moved to $f1$ and $f2$. That is, disassembling $f3$ is beneficial if the left hand side is larger than the right hand side:

$$\frac{c_{s1} + c_{s2} + OH_{f3}}{T_{f3}} + \frac{pay_{f1} + OH_{f1}}{T_{f1}} + \frac{pay_{f2} + OH_{f2}}{T_{f2}} \tag{3.4}$$

$$> \frac{pay_{f1} + c_{s1} + OH_{f1}}{T_{f1}} + \frac{pay_{f2} + c_{s2} + OH_{f2}}{T_{f2}} \tag{3.5}$$

which implies that

$$\frac{c_{s1} + c_{s2} + OH_{f3}}{T_{f3}} > \frac{c_{s1}}{T_{f1}} + \frac{c_{s2}}{T_{f2}} \tag{3.6}$$

That is, $f_3$ might be disassembled and its signals $s_1$ and $s_2$ may be distributed to other frames if (3.6) is true. Assume that $T_{f1} = 50$ms, $T_{f2} = 100$ms. If $T_{f3} = 500$ ms, it is infeasible to disassemble $f_3$:

$$\frac{2 + 2 + 4}{500ms} > \frac{1}{50ms} + \frac{1}{100ms} \tag{3.7}$$

$$\frac{8}{500ms} \not> \frac{3}{100ms} \tag{3.8}$$

However, if we assume that $t_{f3} = 200$ ms, our criterion becomes true which shows that disassembling $f_3$ will decrease the bandwidth utilization:

$$\frac{2 + 2 + 4}{200ms} > \frac{1}{50ms} + \frac{1}{100ms} \tag{3.9}$$

$$\frac{8}{200ms} > \frac{3}{100ms} \tag{3.10}$$

Based on the example, we now generalize the optimality criterion for disassembling frames. Assume that $f$ is a frame with period $T_f$ and signals $s1, \ldots, sk$ and assume that $f1, \ldots, fm$ are frames with $T_{fi} < T_f$ for all $i = 1, \ldots, m$. In addition, consider that $p(sj)$ determines the frame that is assigned to signal $sj$ after disassembling $f$. Then, disassembling $f$ is beneficial if

$$\frac{\sum_{j=1}^{k} c_{sj} + OH_f}{T_f} > \sum_{j=1}^{k} \frac{c_{sj}}{T_{p(sj)}}. \tag{3.11}$$

That is, the bandwidth utilization of $f$ is larger than the bandwidth utilization of the signals of $f$ distributed to other lower-period frames.

The final step of our approach is the schedulability test of the resulting set of packed frames. If the solution is infeasible, frame $f$ with at least two signals and for which the difference between worst-case response time and deadline is the smallest will be found analogous to [15]. Signal $s$ from $f$ with the smallest deadline will be removed and placed in a new frame. Schedulability test will stop when all non-feasible frames have been completely decomposed.

In summary, we derive a heuristic which starts with packing all signals with same periods and then disassembling unnecessary frames according to the proposed optimality criterion.

### 3.4.2 Pseudo Code

We next formulate our method as pseudo code. As stated before, our heuristic is composed of three distinct parts. The first parts aims at constructing a solution that packs signals with similar periods which is called Period Based Heuristic(PBH). The second part deals with destroying unnecessary frames and distributing signals to other available frames. Third part focuses on the feasibility of the proposed solution that is checked with the Audsley algorithm. End of this heuristic is called Advanced Period Based Heuristic(APBH). The pseudo code is given as Algorithm 4.

We define $S$ as the signal set and $F$ as the set of final packed frames. Initially, some preparation steps are needed to group signals with same periods between lines 1 to 5. Signals with same periods are grouped in the signal sets $S_{pn} = \{\ s \in S \mid pn = t_s\ \}$. $P_S$ is set of signal sets $P_S = \{S_{p1}, S_{p2}, .., S_{pn}\}$. In the first part of heuristic, signals in each $S_{pn}$, will be packed in same period frames. Frame sets for each period defined as $F_{p1} = F_{p2} = .. = F_{pn} = \{\}$ and the set of frame sets is $P_F = \{F_{p1}, F_{p2}, .., F_{pn}\}$. Signals in each $S_{pn}$ will be packed in frames with Best Fit Bin Packing heuristic at line 7 [7] and frames will be located in $F_{pn}$ frame sets. Although best fit bin packing heuristic does not cope with bandwidth issues, since periods are fixed in groups, best fit heuristic already gives an efficient result that is usually better than the existing methods as is

**input** : $S$ is list of signals

**output**: $F$ is list of packed frames

Construct signal sets $S_{pn}$ for each unique period where $S_{pn} = \{ s \epsilon S \mid p_s = pn\}$ and p is period

Construct $P_S$, Set of Signal sets $P_S = \{S_{p1}, S_{p2}, .., S_{pn}\}$

For each $S_{pn}$, sort signals in decreasing order of data size

Construct Frame sets $F_{p1} = F_{p2} = .. = F_{pn} = \{\}$

Construct $P_F$, Set of Frame sets $P_F = \{F_{p1}, F_{p2}, .., F_{pn}\}$

**foreach** $S_{pn}$ *in* $P_S$ **do**

    $F_{pn}$ = **BestFitDecreasing**($S_{pn}$)

**end**

Sort $P_F$ increasing order of period

Sort frames in each $F_{pn} \epsilon P_F$ decreasing order of data size

**foreach** *Frame set* $F_{pn}$ *in* $P_F$ **do**

    **foreach** $f \epsilon F_{pn}$ **do**

        $P'_F = P_F$ and $f' = f$ where $f' \epsilon P'_F$

        **foreach** $f_n \epsilon P'_F$ *with smaller period than* $f'$ **do**

            **MoveSignals**(ref $f'$, ref $f_n$)

            **if** $f'$ *is empty* **then**

                **break;**

            **end**

        **end**

        **if** $U(P'_F) < U(P_F)$ **then**

            $P_F = P'_F$

        **end**

    **end**

**end**

**Algorithm 4:** Advanced Period Based Heuristic w/o Schedulability Test

shown in Chapter 4. First part is ended with constructing each frame set $F_{pn}$. Up to this stage we denote our heuristic as Period Based Heuristic (PBH).

In the second part of the heuristic after line 8, the frames with same period signals are evaluated to be disassembled. Basically 3.6 shows that disassembling signals of a frame and moving them to frames with smaller periods may decrease bandwidth utilization of the channel. To determine which frames need to be disassembled, we are iterating over all the frames starting from the lowest data size and highest period at line 11. When considering a frame $f$, firstly we obtain the copies of $P_F$ and frame $f$ as $P'_F$ and $f'$. Then frames with periods smaller than $T_{f'}$ are found and indicated as $f_n$. If some of the signals in $f'$ can fit into the $f_n$, signals are removed from $f'$ and added to $f_n$. The function **MoveSignals** which can be found in Algorithm 6 obtains references of $f'$ and $f_n$, and moves the necessary signals of $f'$ to $f_n$. If $f'$ is empty, $f'$ is removed from the frame set and it is checked whether the bandwidth of the new frame set is lower or not. If the bandwidth utilization of the new solution is lower, we continue with the new frame set $P'_F$ and hence set $P_F = P'_F$. The bandwidth utilization $U(P)$ of the solution is computed with equation (3.2).

Last part of the heuristic consists of schedulability test and necessary operations when the solution is infeasible. Schedulability test of the heuristic follows the procedure in Algorithm 5, whereby Audsley's algorithm [3] and the worst-case response time

calculation in [10] are used.

```
   input : F is list of packed frames
1  if F is feasible then
2      SUCCESS
3  else
4      if All the frames in F contain one signal each then
5          FAILURE
6      else
7          Find the frame in F containing at least 2 signals and which has the least
           difference between the worst case response time and the deadline at the
           lowest priority which has not been assigned. Let the frame be f.
8          Remove the s signal from f with the smallest deadline, and make a new
           frame f' containing only s and add f' to the set F
9      end
10 end
```

**Algorithm 5:** Schedulability Test of APBH.

```
   input : ref f', ref f_n
1  foreach Signal s in f' do
2      if Size of s < Empty size of f_n then
3          Remove signal s from f'
4          Set s' = s and add s' to f_n
5      else
6          continue
7      end
8  end
```

**Algorithm 6: MoveSignals** function.

**Example Case:** We explain the proposed heuristic with an example. We consider 5 signals $s_1$, $s_2$, $s_3$, $s_4$ and $s_5$. They are transmitted over a CAN network with a bit rate $\tau = 125kbit/s$. Size of overhead is $OH = 34$ bits.

Result of the first part of the algorithm give the frame set in Table 3.2. It can be observed that signals with same periods are packed in same frames. Since none of the

| Signal | Period (ms) | Size (bits) |
|---|---|---|
| $s_1$ | 10 | 8 |
| $s_2$ | 50 | 16 |
| $s_3$ | 50 | 16 |
| $s_4$ | 100 | 16 |
| $s_5$ | 100 | 16 |

Table3.1: Signals in the channel

unique periods are eliminated, $gcd(T)$ equals to 100 ms.

| Frame | Signals | Period(ms) | Size(bits) | Utilization |
|---|---|---|---|---|
| f1 | s1 | 10 | 8 | 3,36% |
| f2 | s2, s3 | 50 | 32 | 1,06% |
| f3 | s4, s5 | 100 | 32 | 0,53% |
| | | | $U(F)$**:** | 4,94% |

Table3.2: Frames after first part of the heuristic

Second part of the heuristic tries to disassemble $f3$ if distributing signals $s3$ and $s4$ to other frames gives a better utilization. Since the total bus utilization decreases as shown in Table 3.3, destroying $f3$ is more efficient.

| Frame | Signals | Period(ms) | Size(bits) | Utilization |
|---|---|---|---|---|
| f1 | s1 | 10 | 8 | 3,36% |
| f2 | s2, s3, s4, s5 | 50 | 64 | 1,57% |
| | | | $U(F)$: | 4,93% |

Table3.3: Frames after second part of the heuristic

# CHAPTER 4

# EXPERIMENTS

## 4.1 Experimental Setup

We have performed many experiments based on state-of-art studies and our approach. The main inputs of our experiments are the bus load, the CAN bit rate, period distribution of signals, deadlines of signal, size distribution of signals, and the maximum size of packed frames. Signal sets which are used for the experiments are generated randomly with different periods and sizes. For instance; a generated signal set contains signals between size of 1 to 32 bits but mostly 1 to 10 bits. Another example is that periods of signals are spread between 10ms to 1000ms but signals with period 10ms to 100ms constitute the majority of the signal set. A detailed description of the signal sets is given in Section 4.2.

State-of-art algorithms and our approach are implemented on the same software platform such that we are able to use the same generated signal sets as the input of all stated approaches. Output of packing algorithms give a frame set with properties of new bus load and schedulability. The algorithmic complexity of the different methods directly influences the required run-time.

Experiments are performed on a software we implemented called CANBenchmark which is developed in the .NET Framework with the C# programming language and compiled with Microsoft Visual Studio 2010. CANBenchmark Software basically imports CAN signals in different file formats such as .dbc, .xml and .csv. It evaluates the schedulability of the signal set for a defined bus bit rate. It calculates bus utilization for CAN and CAN-FD bus bit rates according to WCTT equations (3.2).

We note that the resolution of signal sizes is given in Byte in the existing methods. Nevertheless, real applications require a signal size resolution of bits. Accordingly, the CANBenchmark software supports bitwise signals.

A screenshot of the main window of the CANBenchmark software is shown in Fig. 4.1.

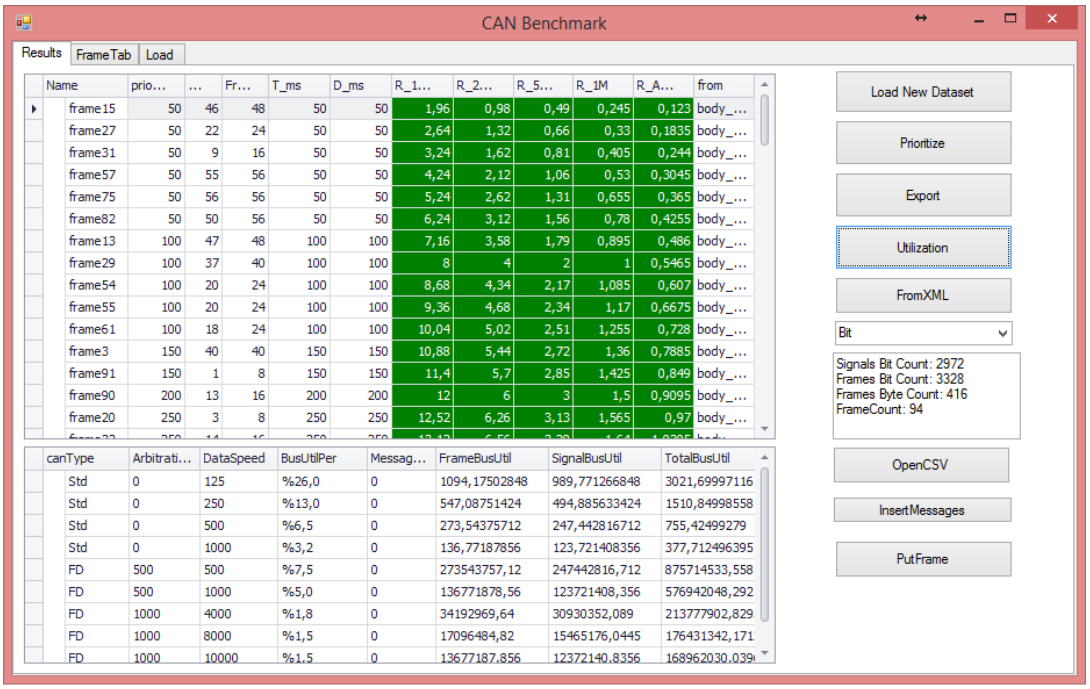



Figure 4.1: CANBenchmark Software Main Window

The tab in Fig. 4.1 is used to import signal sets from files which are generated before. The main focus of this thesis is frame packing which is accomplished in the "FrameTab". "FrameTab" provides a combo-box with the implemented algorithms. Signals are then packed according to the selected algorithm. The "FrameTab" is shown in Fig. 4.2.

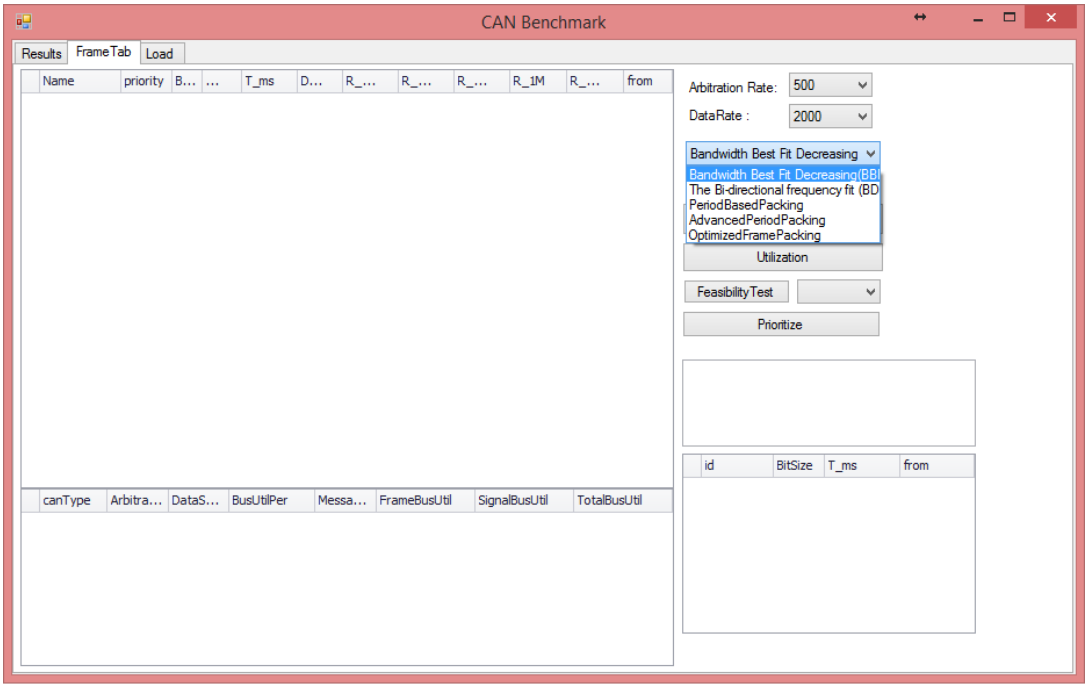



Figure 4.2: CANBenchmark Software FrameTab with Algorithms

To pack signals, the "Pack Signals" button in the "FrameTab" should be clicked. Packed frames are then shown on the main list of the "FrameTab" window. The main list of packed frames is shown with number "1" in Fig. 4.3. The second list which is shown as number 2 consists of the utilization results of the resulting packed frame set according to different bus bit rates including CAN and CAN-FD configurations. Finally the list shown with number 3 presents the signals of each selected frame from List 1. In the Fig. 4.3, the selected frame on List 1 consists of three signals with id's 398, 624 and 510. Signal sizes are consequently 32, 21 and 11 bits, therefore the frame size is 64 bits.

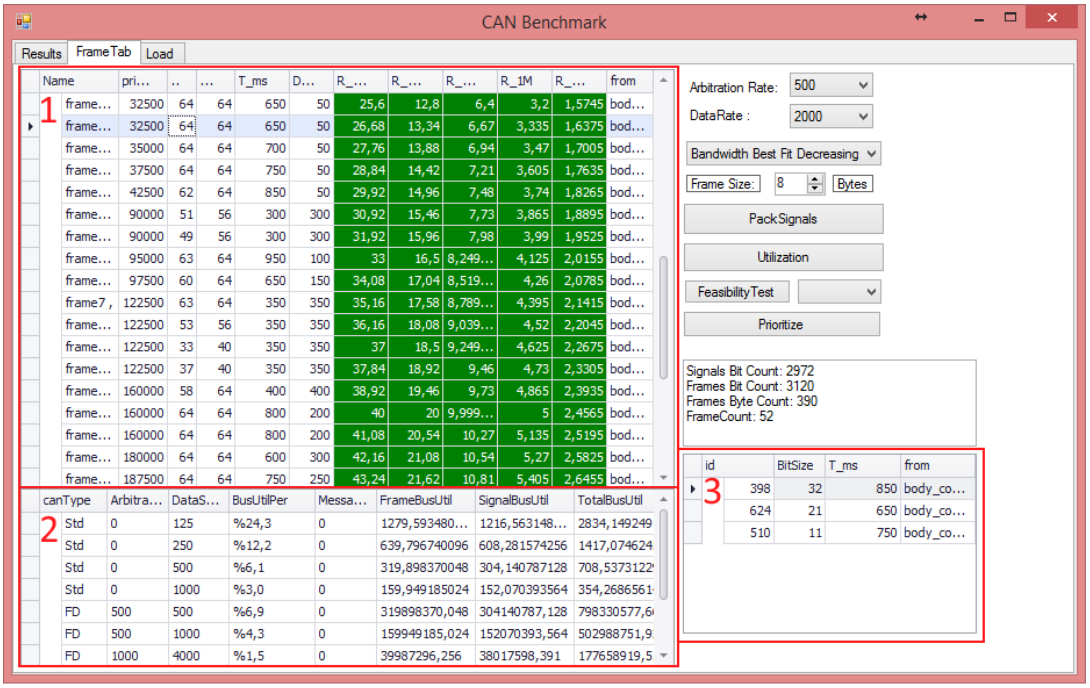



Figure 4.3: CANBenchmark Software FrameTab after Packing

1SpF, BBFd, BDFF, IFPH and our approaches PBH and APBH are used to obtain frame packing results of 350 signal sest. In the subsequent sections, the conducted experiments are stated in detail.

## 4.2 Generated Signal Sets

In this section, signal set configurations with different properties are explained in detailed. Each signal group stated in this section consists of 20 randomly generated signal sets. Signal sets are generated with the NETCARBENCH [5] software which is used in the design and configuration of CAN communication systems. NETCARBENCH generates sets of signals basaed on a user defined configuration. Configuration parameters consists of pre-defined bus load, period distribution, size distribution and node number of a signal set desired to be generated.

Signals are generated according to user defined periods and sizes. The interval of periods is chosen between 5 ms to 1000 ms and the interval of sizes is chosen between

1 bit to 64 bits for the Signal Groups we generated. Mostly, signals with size up to 32 bits are used for the experiments because these are the most relevant for the automotive domain. We have referenced the benchmark data reported by the Society of Automotive Engineers (SAE) [8] to evaluate what type of signals a vehicle network may have.

Histogram figures in each signal group indicate distribution of average number of signals according to periods and sizes of 20 signal sets generated with same input parameters by NETCARBENCH.

### 4.2.1 Signal Group 1: Small Size in 1-32 bits and Small Periods (SSLP)

The period and size distribution of this signal group is shown in Fig. 4.4 and 4.5, respectively. This signal group consists of mostly small signals (smaller than 10 bit) with small periods (mostly smaller than 100 ms).

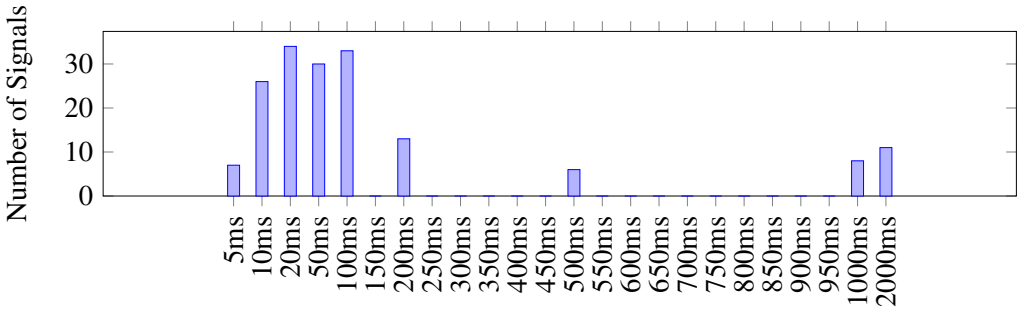



Figure 4.4: Average period distribution of signals in Signal Group 1

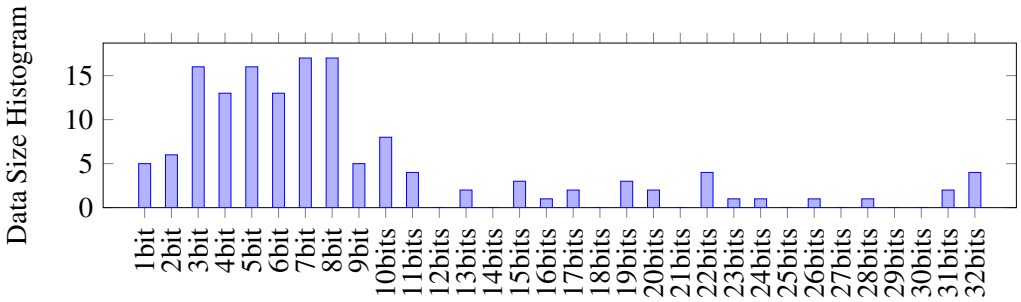



Figure 4.5: Average size distribution of signals in Signal Group 1

### 4.2.2 Signal Group 2 : Smaller Size in 1-32 bits and Uniform Period (SSUP)

Signal Group 2 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.6 and 4.7, respectively. Average number of signals is 1870 and average number of total bits is 17000 in Signal Group 2. Signals up 10 bits are most frequent to be packed in frames.

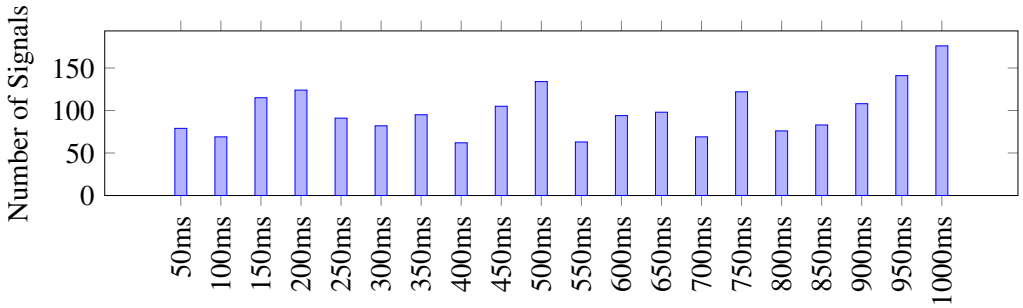



Figure 4.6: Average period distribution of signals in Signal Group 2

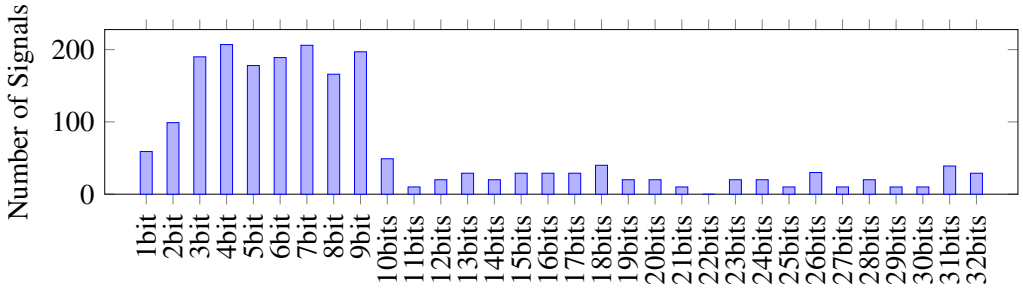



Figure 4.7: Average size distribution of signals in Signal Group 2

### 4.2.3 Signal Group 3: Medium Size in 1-32 bits and Uniform Periods (MSUP)

Signal Group 3 consists of 20 signal sets. Each signal set has signals with a same distribution of size and periods indicated in Fig. 4.8 and 4.9, respectively. Average number of signals is 1850 and average number of total bits is 29000 in Signal Group 3. Signals between 10 to 20 bits are more difficult to be packed compared to Signal Group 2.

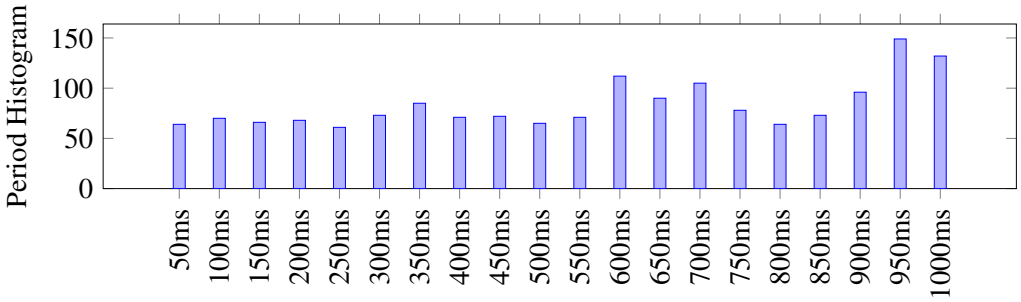



Figure 4.8: Average period distribution of signals in Signal Group 3

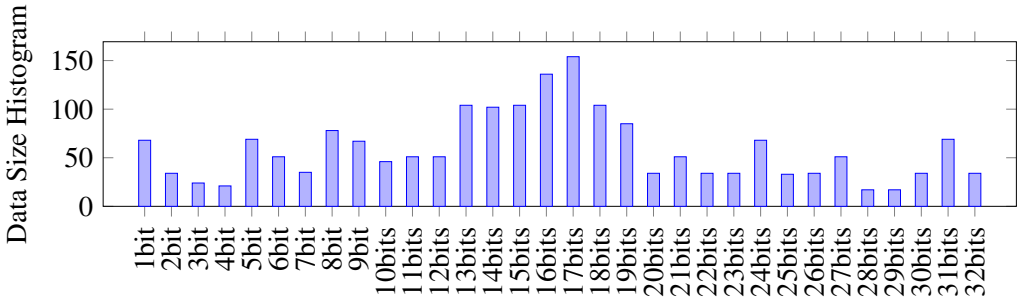



Figure 4.9: Average size distribution of signals in Signal Group 3

### 4.2.4 Signal Group 4: Larger Size in 1-32 bits and Uniform Periods (LSUP)

Signal Group 4 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.10 and 4.11, respectively.. Average number of signals is 1860 and average number of total bits is 34600 in Signal Group 4. Signals between 20 to 32 bits are more difficult to be fit in a 64 bit CAN Message.

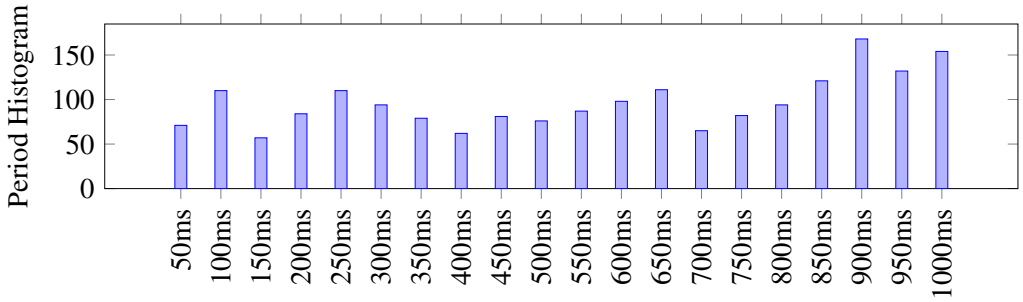



Figure 4.10: Average period distribution of signals in Signal Group 4

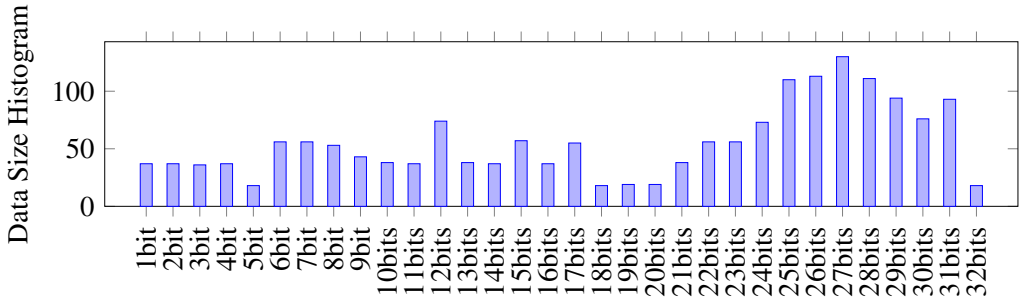



Figure 4.11: Average size distribution of signals in Signal Group 4

### 4.2.5 Signal Group 5: Uniform Size in 1-32 bits and Lower Periods (USLP)

Signal Group 5 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.12 and 4.13, respectively. Average number of signals is 1650 and average number of total bits is 27200 in Signal Group 5. This signal set with smaller periods results in a higher bus load.

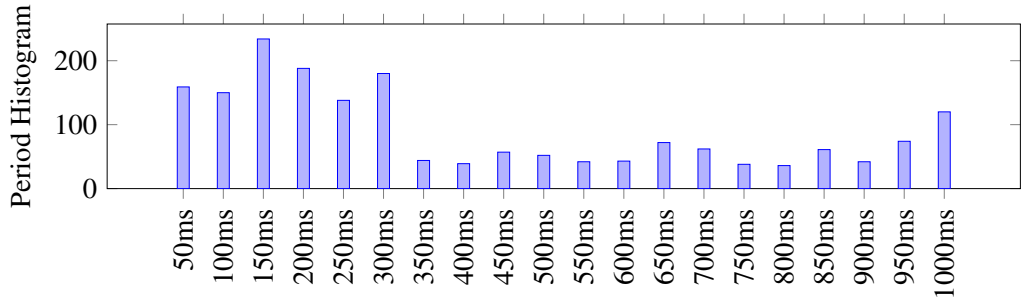



Figure 4.12: Average period distribution of signals in Signal Group 5

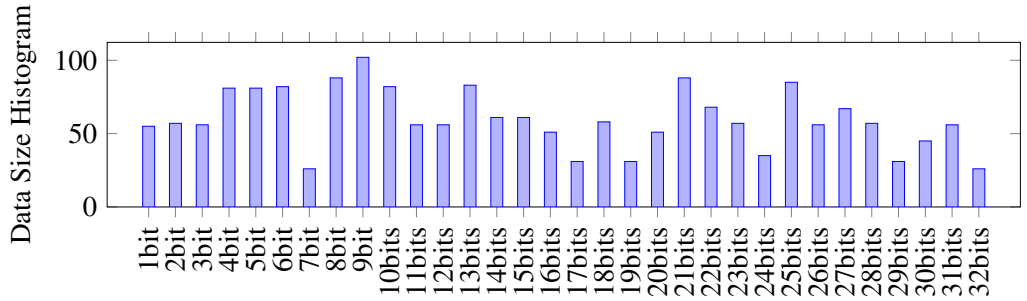



Figure 4.13: Average size distribution of signals in Signal Group 5

### 4.2.6 Signal Group 6: Uniform Size in 1-32 bits and Medium Periods (UPMP)

Signal Group 6 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.14 and 4.15, respectively. Average number of signals is 1700 and average number of total bits is 27500 in Signal Group 6.

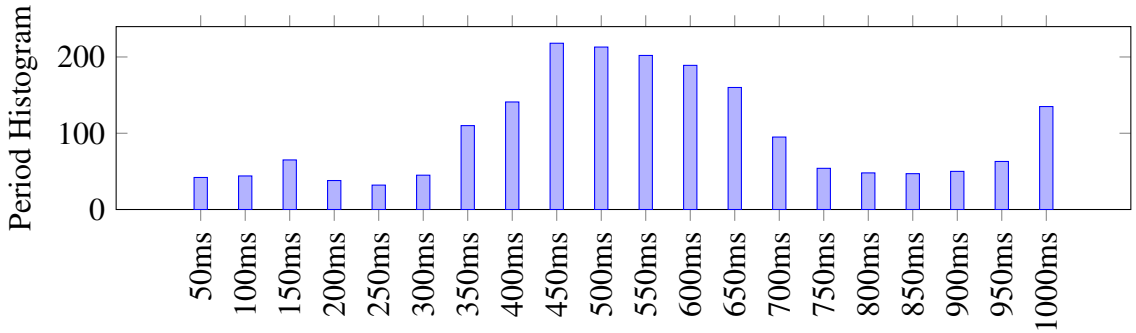



Figure 4.14: Average period distribution of signals in Signal Group 6

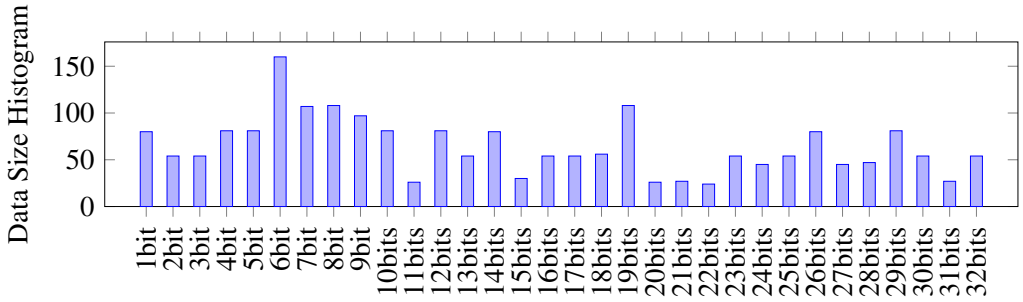



Figure 4.15: Average size distribution of signals in Signal Group 6

### 4.2.7 Signal Group 7: Uniform Size in 1-32 bits and Higher Periods (USHP)

Signal Group 7 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.16 and 4.17, respectively. Average number of signals is 1150 and average number of total bits is 17500 in Signal Group 7.

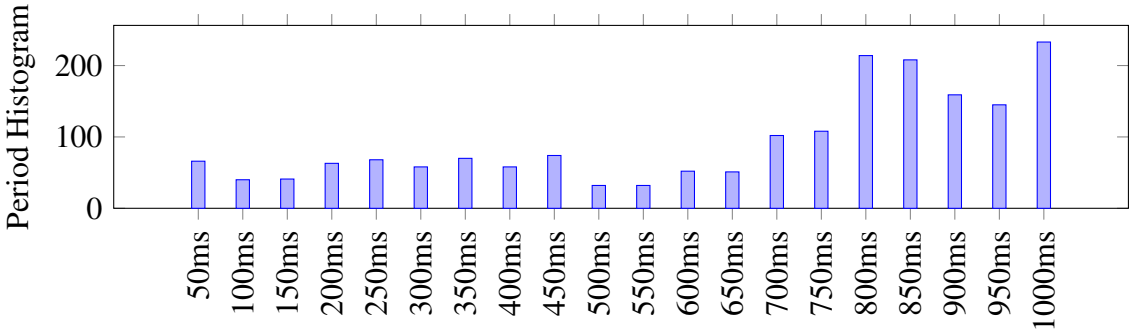



Figure 4.16: Average period distribution of signals in Signal Group 7

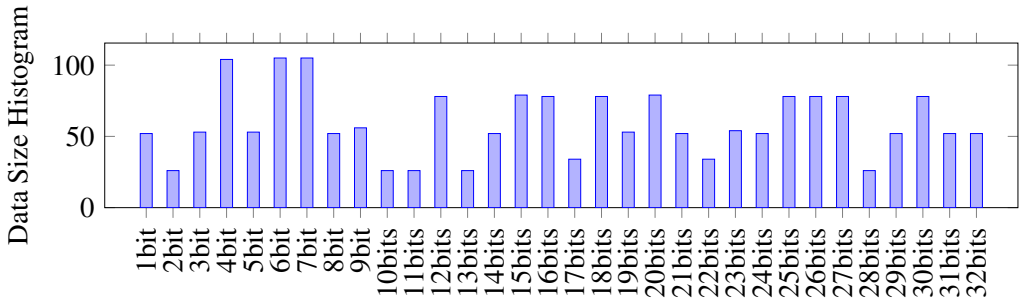



Figure 4.17: Average size distribution of signals in Signal Group 7

### 4.2.8 Signal Group 8: Uniform Size in 1-16 bits and Uniform Periods (USUP)

Signal Group 8 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.18 and 4.19, respectively. Average number of signals is 1910 and average number of total bits is 16600 in Signal Group 8. Signals with up to 16 bits size is most easiest signal set to be packed.

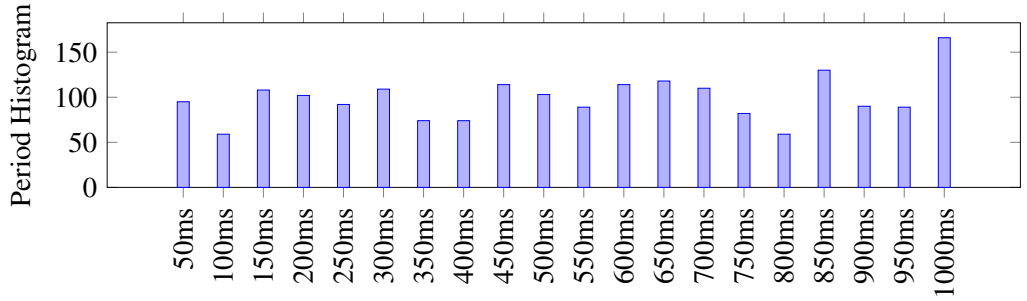



Figure 4.18: Average period distribution of signals in Signal Group 8

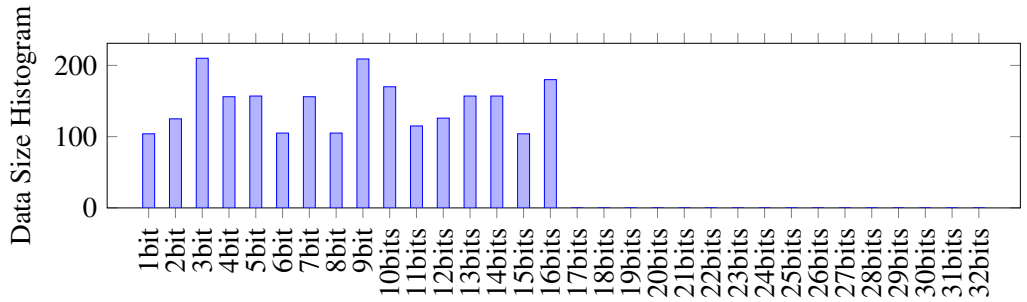



Figure 4.19: Average size distribution of signals in Signal Group 8

### 4.2.9 Signal Group 9: Uniform Size in 1-32 bits and Uniform Period (USUP)

Signal Group 9 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.20 and 4.21, respectively. Average number of signals is 1150 and average number of total bits is 1700 in Signal Group 9.

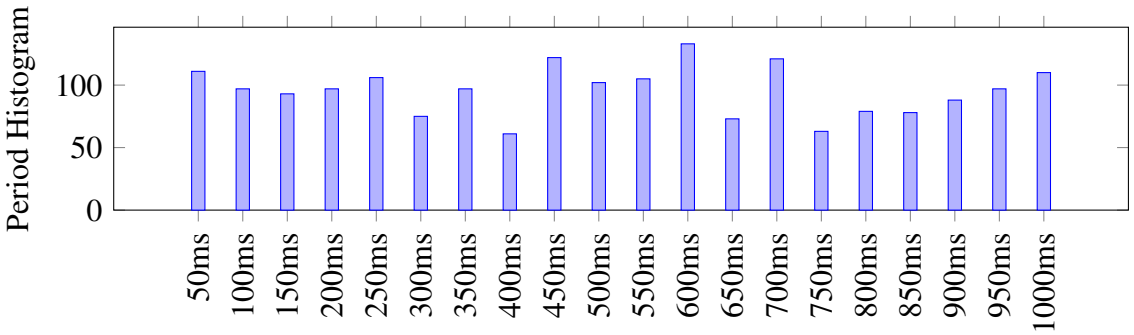



Figure 4.20: Average period distribution of signals in Signal Group 9

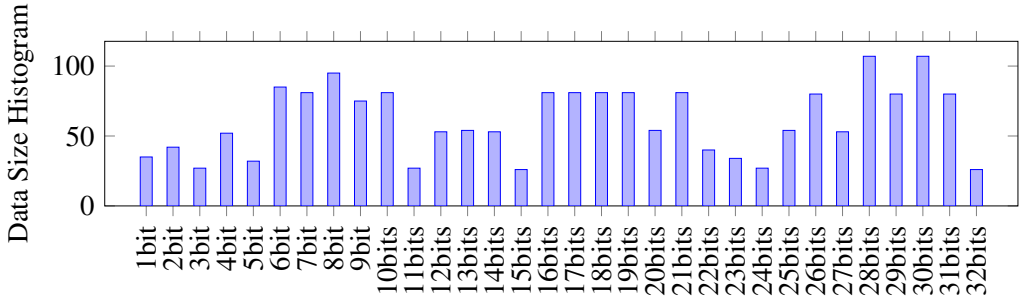



Figure 4.21: Average size distribution of signals in Signal Group 9

### 4.2.10 Signal Group 10: Uniform Size in 1-64 bits and Uniform Period (USUP)

Signal Group 10 consists of 20 signal sets. Each signal set has signals with a distribution of size and periods indicated in Fig. 4.22 and 4.23, respectively. Average number of signals is 1696 and average number of total bits is 53400 in Signal Group 10. Signals with up to 64 bits size are more difficult to pack.

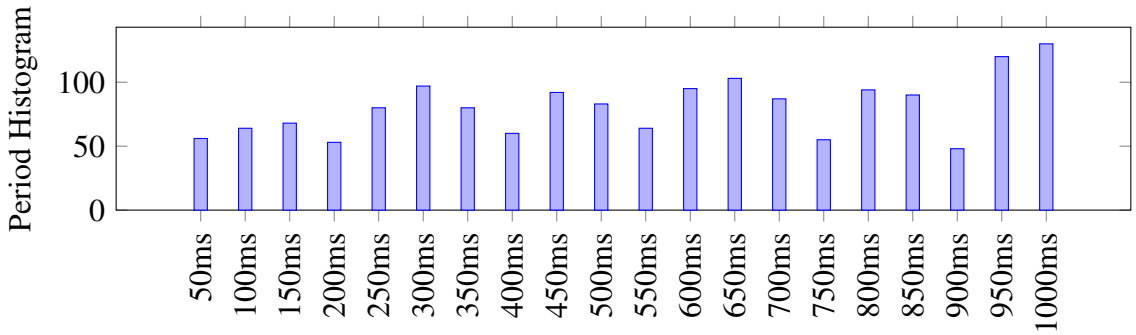



Figure 4.22: Average period distribution of signals in Signal Group 10

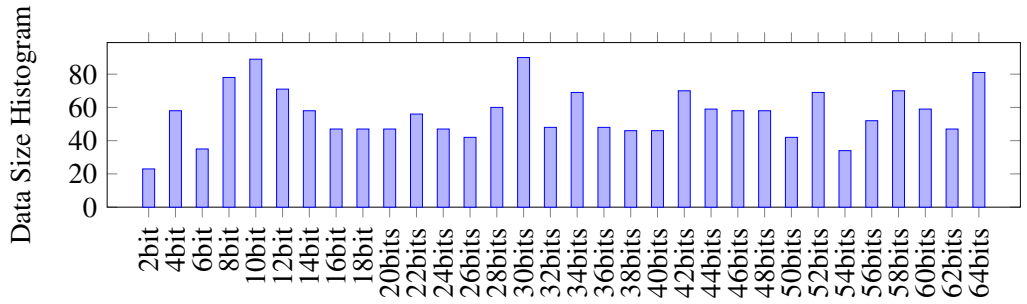


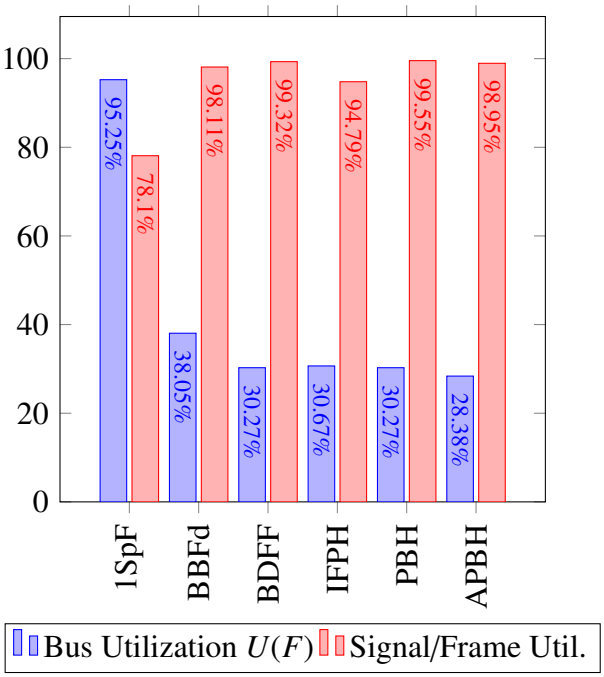

Figure 4.23: Average size distribution of signals in Signal Group 10

## 4.3 Results For CAN

Our approach and the previous studies are compared with regard to the bandwidth consumption of the resulting set of frames. Benchmark results and schedulability of the solutions are obtained from CANBenchmark software. Experiments are performed on the previously described randomly generated signal sets. The bus utilization on the bar chart presented in the experiment tables is the average bus utilization of the frame packing solution of 20 different signal sets.

| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 1: Small Size in 1-32 bits and Small Periods (SSLP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

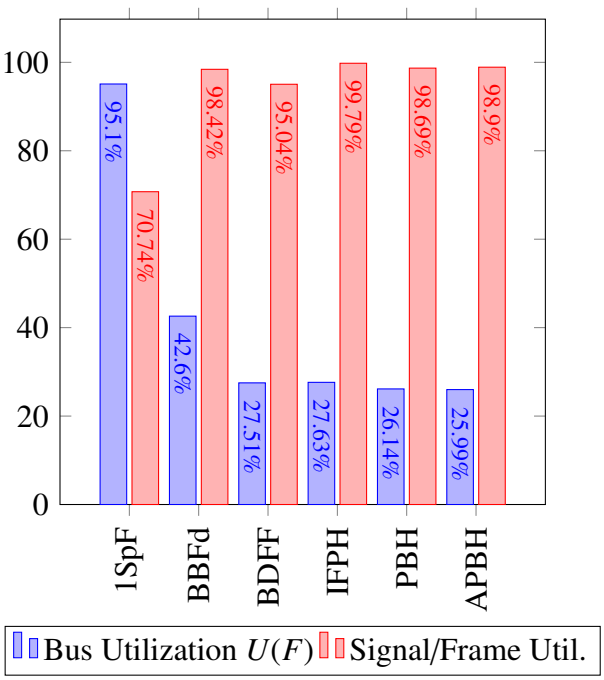
Table4.1: Experiment Signal Group 1




Experiments with Signal Group 1 shows that even Period Based Packing (PBH) which is the first part of our approach results in a better utilization than all the other approaches. In addition, Advanced Period Based Heuristic (APBH) gives the best result.

| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 2 : Smaller Size in 1-32 bits and Uniform Period (SSUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.2: Experiment Signal Group 2

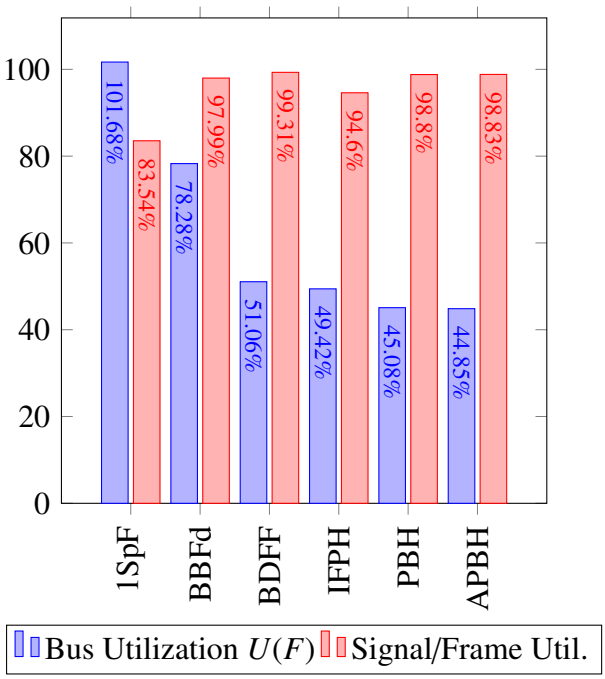



It can be observed by experiments with Signal Group 2 that except BBFd and 1SpF

approach, other heuristics use the period property of the signals more efficiently which causes better bandwidth utilization. Again, our methods PBH and APBH give the best results.

| **Input Signal Set Properties** | **Value** |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 3: Medium Size in 1-32 bits and Uniform Periods (MSUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

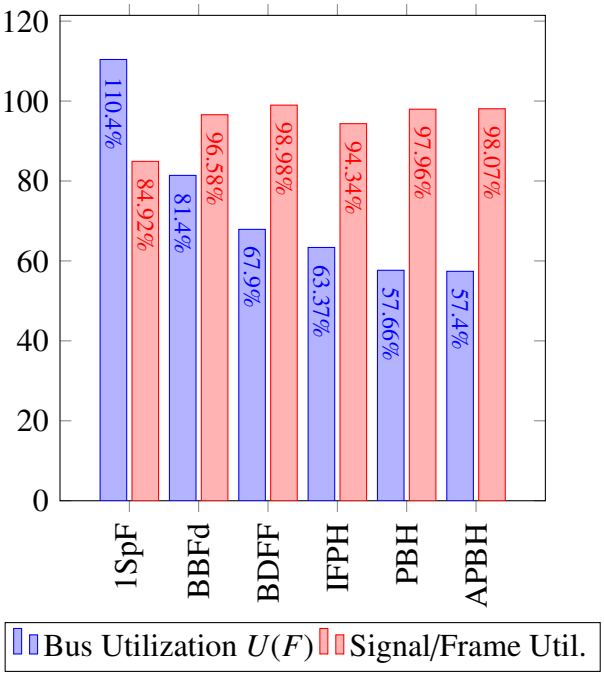
Table4.3: Experiment Signal Group 3




IFPH heuristic has a larger bandwidth utilization than BDFF with Signal Group 2 which has smaller sized signals between 1 to 32 bits. However, experiments with Signal Group 3 gives a different result. Here, the bandwidth utilization of IFPH is better than BDFF. Polzlbauer [14] states that IFPH performs well for a wide range of sizes (up to 32 bits). It is observed that IFPH has better results especially among signals with sizes between 10 bits to 32 bits. Nevertheless, our approaches BPH and APBH significantly improve on IFPH for the medium size signals.

| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 4: Larger Size in 1-32 bits and Uniform Periods (LSUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

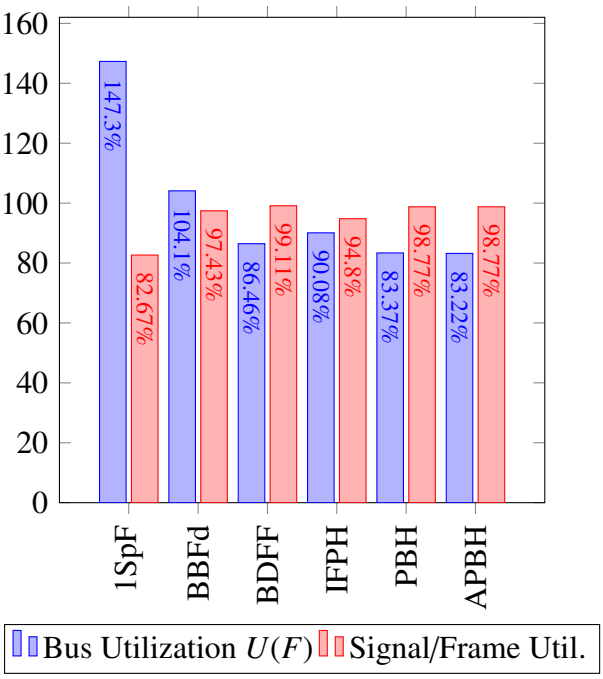
Table4.4: Experiment Signal Group 4




Except our approaches PBH and APBH, IFPH has the best results with Signal Group 3 and 4 which contains signals with sizes mostly between 10 to 32 bits in 1 to 32 bits size scale. It is observed that IFPH does not achieve better bandwidth utilization than BDFF with uniform size distribution almost in all cases. Moreover, the improvement of our methods increases with increasing signal size.

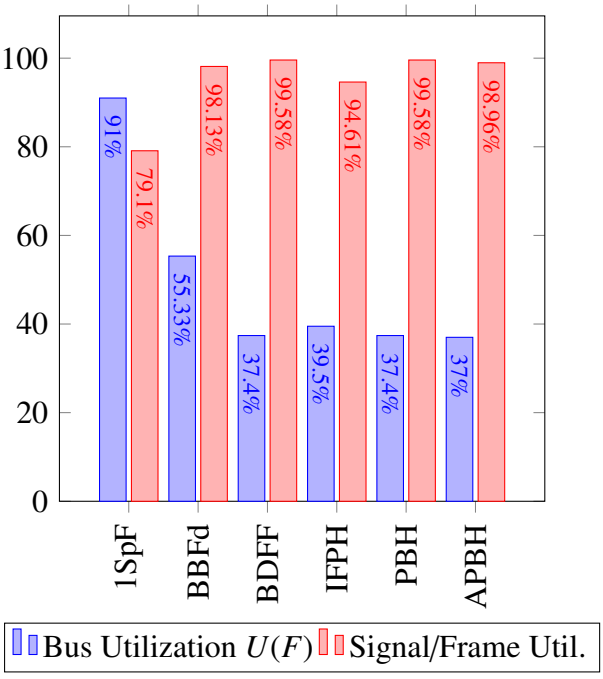
| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 5: Uniform Size in 1-32 bits and Lower Periods (USLP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.5: Experiment Signal Group 5



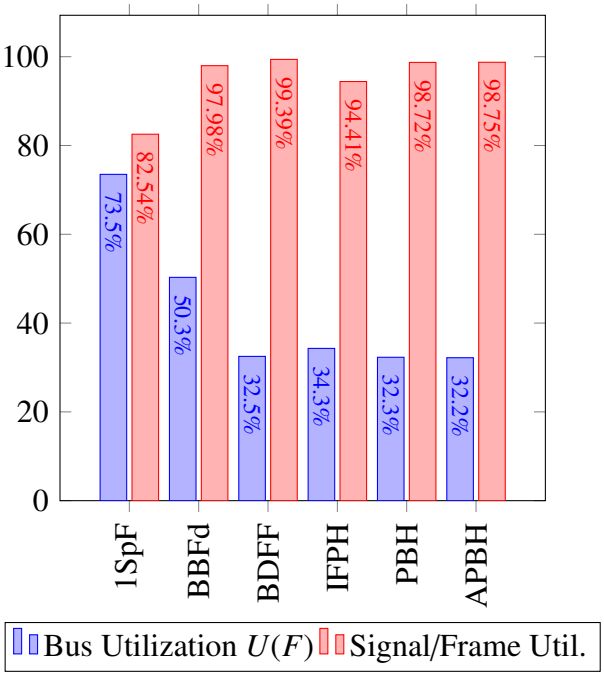

| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 6: Uniform Size in 1-32 bits and Medium Periods (UPMP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.6: Experiment Signal Group 6



| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 7: Uniform Size in 1-32 bits and Higher Periods (USHP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

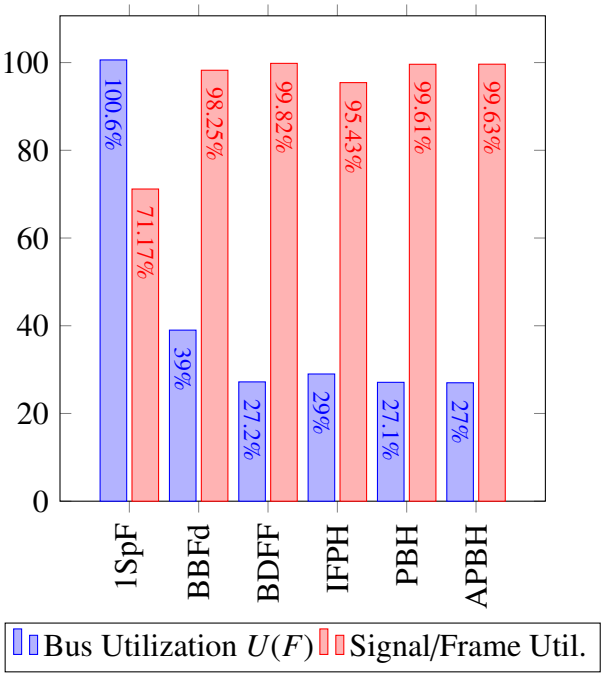
Table4.7: Experiment Signal Group 7




Signal Group 5, 6 and 7 have signals with uniform distribution sizes between 1 to 32 bits but different distributions over period. It is observed that distribution over period is not affecting efficiency of the algorithms on bandwidth utilization. Again, in all cases, our methods achieve the smallest bandwidth utilization.

| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 8: Uniform Size in 1-16 bits and Uniform Periods (USUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

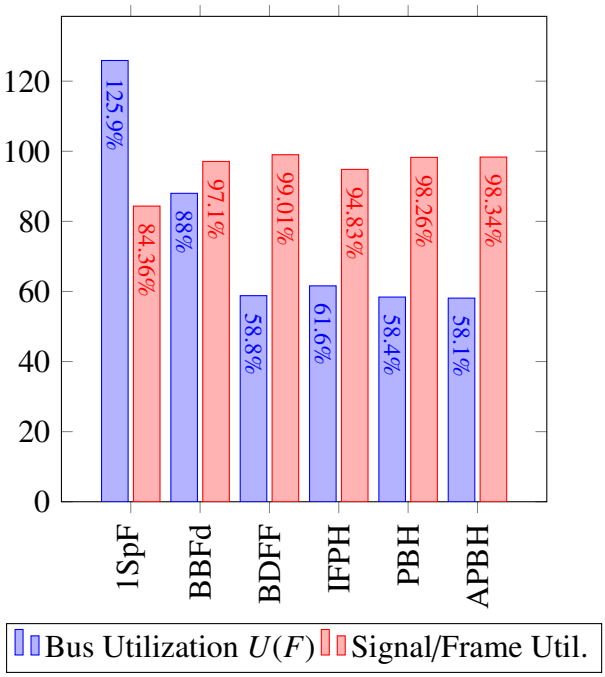
Table4.8: Experiment Signal Group 8



Range of signals between 1 to 16 bits, are the easiest group to be packed into frames with maximum size of 64 bits. Small sized signals are fitting into empty spaces of frames much easier than larger sized signals. Our approach APBH gave the best bandwidth utilization over signals with uniform size distribution in the range of 1 to 16 bits.

| **Input Signal Set Properties** | **Value** |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 9: Uniform Size in 1-32 bits and Uniform Period (USUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

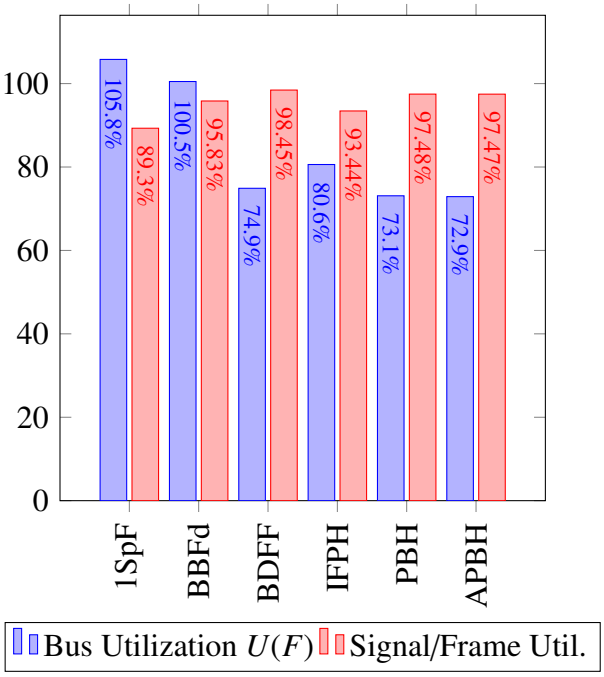
Table4.9: Experiment Signal Group 9




Most relevant signal sizes in automotive domain are between 1 to 32 bits. We experimented the Signal Group 9 with uniform size distribution over 1 to 32 bits and uniform period distribution over 50 ms to 1000 ms. Most of the signal groups we generated for experiments, have a range of size between 1 to 32 bits but have different size distributions. Although there is a very small difference between APBH and BDFF, our approach APBH gives the best result in experiments with Signal Group 9.

| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 10: Uniform Size in 1-64 bits and Uniform Period (USUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.10: Experiment Signal Group 10

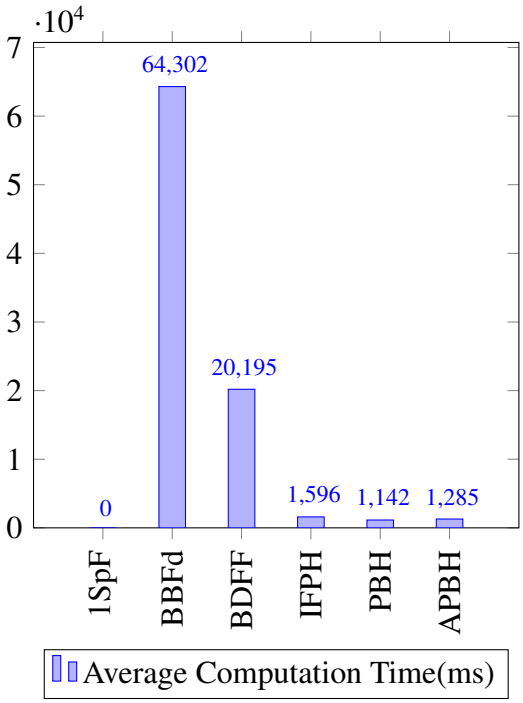



Although size more than 32 bits are not highly relevant for the automotive domain, we have had an experiment with a uniform size distribution between 1 to 64 bits. It is observed that APBH resulted the best bandwidth utilization.

Computation time differs for all approaches to packing problem. Since the problem is NP Hard[16], heuristics proposed spend a significant time that we observed in our experiments. For Signal Group 3, computation time of algorithms is indicated in following graph.

| Input Signal Set Properties | Value |
|---|---|
| **Bit rate** | 500kbit/s |
| **Signal Group** | Signal Group 3: Medium Size in 1-32 bits and Uniform Periods (MSUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.11: Computation Time Signal Group 3




IFPH and our approach have similar computation time which is significantly smaller than BBFd and BDFF. APBH is higher than PBH, since APBH contains an additional part after PBH which is indicated in Frame Packing Algorithms sections detailed. Consequently, our approach APBH has the smallest computation time while achieving the best results.

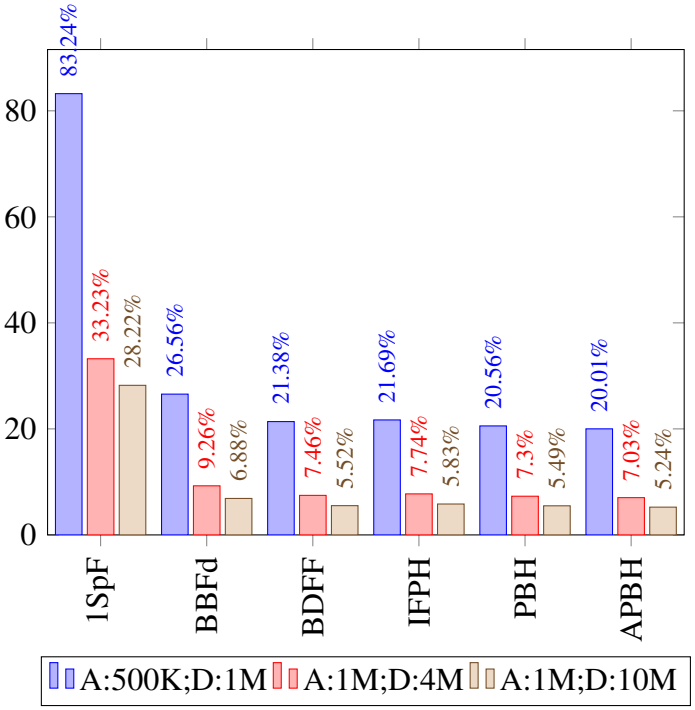
## 4.4 Results For CAN-FD

Our approach and the previous studies are compared with regard to the bandwidth consumption of the resulting set of frames using the same signal sets as in the previous section. Different from the CAN results, the bandwidth utilization with different bit rates is shown for CAN-FD results. Label of the histograms contains the bit rate configuration of the channels, for instance; D:500;A:1K means that data bit rate is 500Kbit/s and arbitration bit rate is 1000Kbit/s of proposed CAN-FD network. Three different bit rate configurations are selected to cover small and large bit rates of CAN-FD. Let us define Data bit rate as $\tau_D$ and arbitration bit rate as $\tau_A$. Three different bit rate configurations used in the experiments are ($\tau_A$ = 500Kbit/s, $\tau_D$ = 1000Kbit/s),

($\tau_A$ = 1000Kbit/s, $\tau_D$ = 4000Kbit/s) and ($\tau_A$ = 1000Kbit/s, $\tau_D$ = 10000Kbit/s).

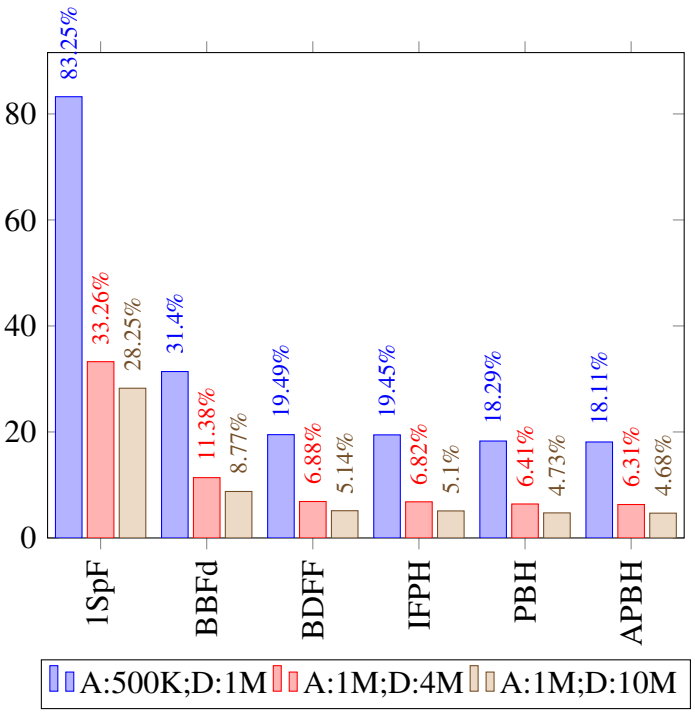
| Input Signal Set Properties | Value |
|---|---|
| **Signal Group** | Signal Group 1: Small Size in 1-32 bits and Small Periods (SSLP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.12: CANFD Experiment Signal Group 1




| Input Signal Set Properties | Value |
|---|---|
| **Signal Group** | Signal Group 2 : Smaller Size in 1-32 bits and Uniform Period (SSUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

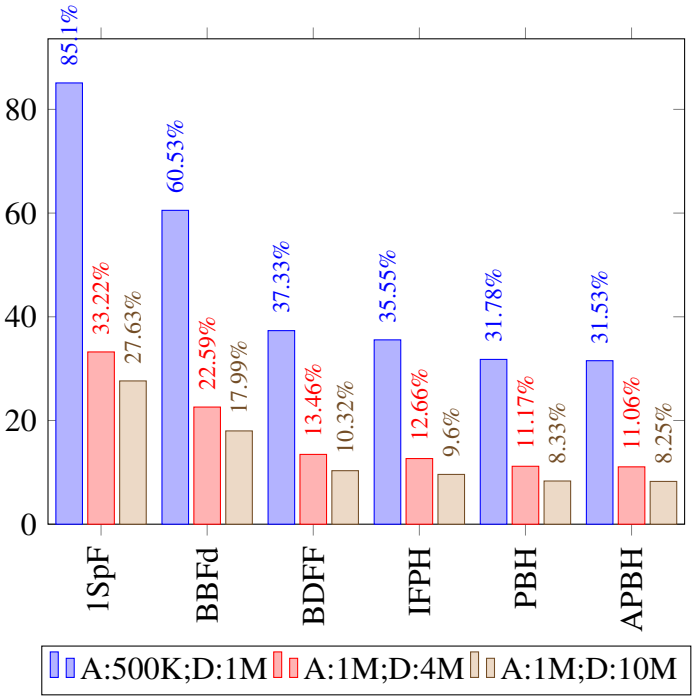
Table4.13: CANFD Experiment Signal Group 2




Experiments with CAN-FD on bandwidth utilization shows that our approach has the lowest bandwidth utilization for all types of input signal groups. It is further observed that IFPH is better than BBFD for smaller sized signals among 1 to 32 bits.

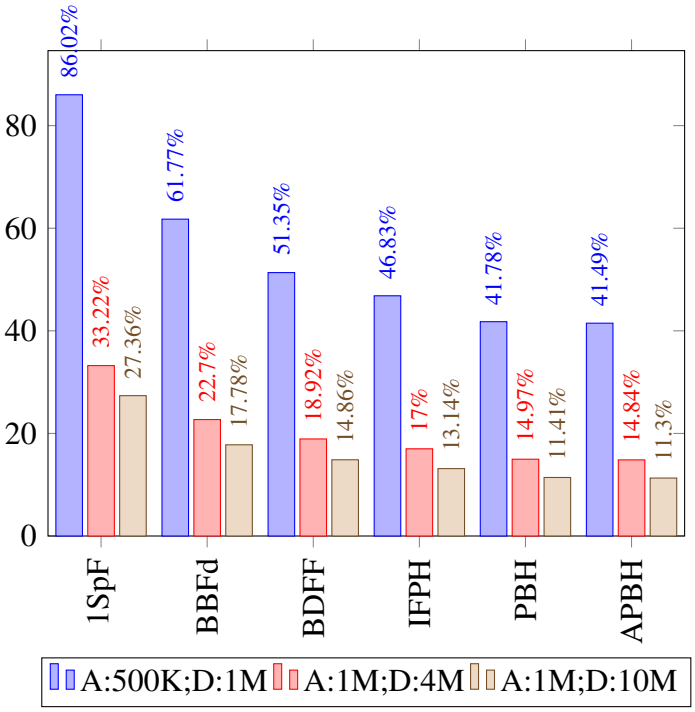
| Input Signal Set Properties | Value |
|---|---|
| **Signal Group** | Signal Group 3: Medium Size in 1-32 bits and Uniform Periods (MSUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.14: CANFD Experiment Signal Group 3




| Input Signal Set Properties | Value |
|---|---|
| **Signal Group** | Signal Group 4: Larger Size in 1-32 bits and Uniform Periods (LSUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

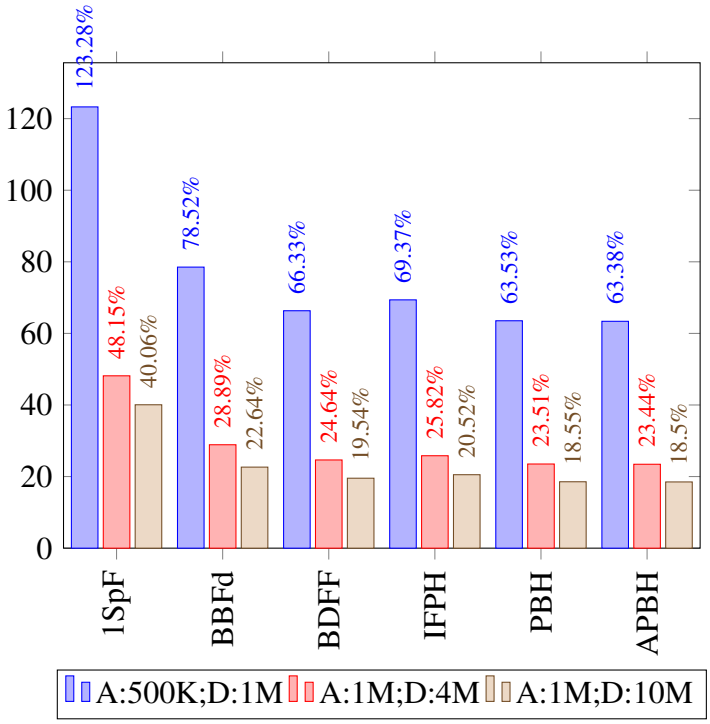
Table4.15: CANFD Experiment Signal Group 4




Like in CAN results, Signal Group 3 and 4 resulted as IFPH has a lower utilization than BDFF. For the two signal groups APBH has the lowest bandwidth utilization for CAN and also CAN-FD experiments.

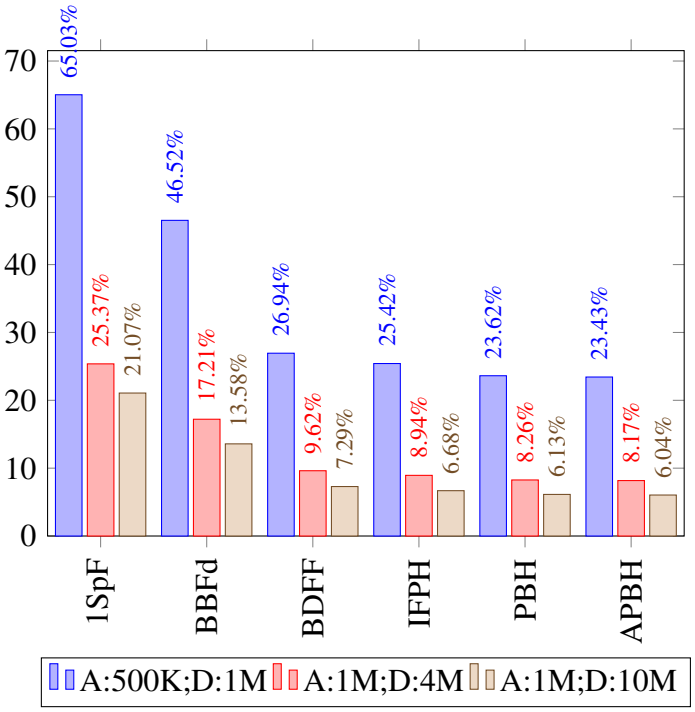
| Input Signal Set Properties | Value |
|---|---|
| **Signal Group** | Signal Group 5: Uniform Size in 1-32 bits and Lower Periods (USLP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.16: CANFD Experiment Signal Group 5



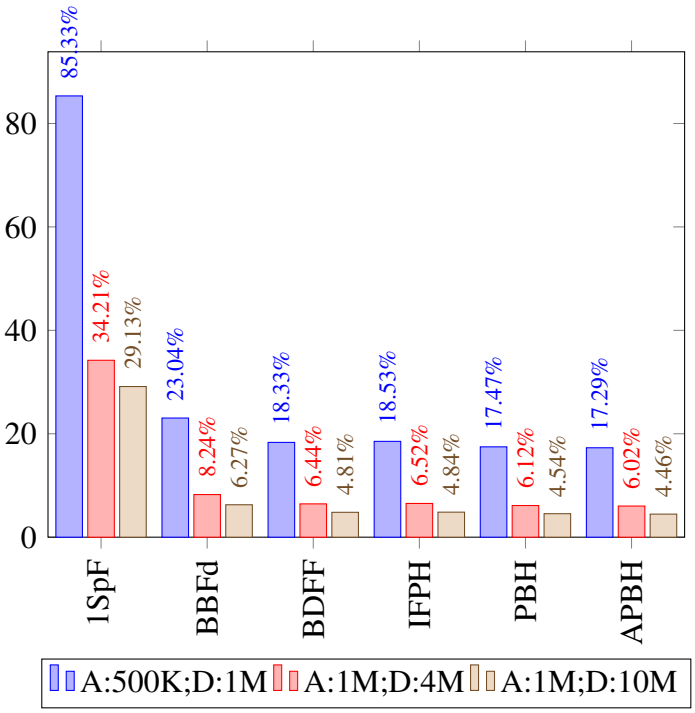

| Input Signal Set Properties | Value |
|---|---|
| **Signal Group** | Signal Group 7: Uniform Size in 1-32 bits and Higher Periods (USHP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.17: CANFD Experiment Signal Group 7



| Input Signal Set Properties | Value |
|---|---|
| **Signal Group** | Signal Group 8: Uniform Size in 1-16 bits and Uniform Periods (USUP) |
| **Deadline** | Equal to period |
| **Number of Station** | 1 |
| **Number of Generated Signal Sets** | 20 |

Table4.18: CANFD Experiment Signal Group 8




## 4.5 Discussion

Beside bandwidth utilization, signal/frame utilization provides a benchmark for how signals in a signal set fit into the frames that resulted after packing operation. Packing signals with same periods into the frames always decreases bandwidth utilization. Constructing fully utilized frames decreases bus utilization independently from the period. However, if there are signals with different periods, putting signals with higher period into frames with lower period increases the bandwidth utilization at the same time signal/frame utilization increase.

From the experimental results, it can be observed that some algorithms with higher signal/frame utilization lead to a lower bandwidth utilization. This seemingly counter-intuitive observation shows how effective the algorithms use the period property of the signals according to the bandwidth problem.

[12] (BBFd) states that the proposed packing procedure depends on a bin packing solution under consideration of the bandwidth consumption. Our experimental results rather show that although signal/frame utilization of BBFd is not lower than %97

which indicates on a good packing operation, the bandwidth utilization is always lower than the other algorithms. Hence, bin packing is not the primary task of frame packing. On the other hand, IFPH [14] results in the best bus utilization among the state of art studies while showing a lowest signal/frame utilization in most of the cases. Our experiments clearly show that the signal period is the major property of the packing problem on CAN and CAN-FD.

In addition, the signal size has an important role in packing problem. Since signals with smaller size can fit into the empty spaces easier, packing efficiency becomes better as seen in the experiments Table 4.10, Table 4.8. In the experiment with Signal Group 8 (Table 4.8), the bandwidth utilization of BBFd compared to 1SpF decreases by a ratio of %61. In contrast, in experiment with Signal Group 9, decreasing ratio is %39. Signal Group 8 has uniform distributed signals between 1 to 16 bits which has smaller sizes than group 9. Sizes up to 32 bits are the most relevant ones in automotive domain. Our approach performs well for the whole range of signal sizes from 1 to 64 bits. Especially Signal Groups 2, 3 and 4 are generated to obtain signal sets with different size distributions. Finally, we note that the run-time of our method is smaller compared to the existing approaches.

# CHAPTER 5

# CONCLUSION

The subject of this thesis is the frame packing of signals for the CAN protocol and its extended version CAN-FD (CAN with Flexible Data Rate). As the main observation of our study, it is found that the signal period is the most relevant signal property for signal packing. That is, it is most beneficial to pack signals with the same period into the same frame. Based on this observation, it was possible to propose two new frame packing algorithms that outperform the frame packing algorithms in the existing literature: our Period Based Heuristic (PBH) uses bin packing in order to assemble frames from signals with the same period; our Advanced Period Based Heuristic (APBH) is an improvement of PBH that tries to disassemble short frames and put their signals into empty spaces of other frames in order to reduce the bandwidth utilization. Our experimental results with a large number of signal sets with different properties show that APBH is always better in bandwidth utilization and at the same time comes with the lowest computation time among existing frame packing approaches.

A systematic comparison of the existing approaches for frame packing on CAN or CAN-FD has not been performed before. In this thesis, all existing frame packing algorithms as well as our new algorithms are implemented in the form of a software tool CANBenchmark. The existing methods include Bandwidth-Best-Fit decreasing (BBFd) [12], Bi-directional Frequency Fit (BDFF) [15], Improved Frame Packing Heuristic (IFPH) [14]. Experiments with different signal sets are carried out to evaluate the performance of the different algorithms. To this end, specific signal groups with different signal properties such as signal size distribution and signal period distribution are randomly generated and all frame packing methods are applied to these

signal sets for both CAN and CAN-FD.

We observed that the signal size distribution has an important influence on the packing problem. For example, it turns out that BDFF gives better results than IFPH for small signal sizes, while IFPH performs better for medium and large signal sizes. Nevertheless, our methods PBH and APBH show the best performance in all cases. It is also observed that the distribution of signal periods has a smaller impact on the results of the algorithms than the size distribution.

In addition to the bandwidth utilization, we also compare the Signal/Frame utilization for CAN for the different frame packing approaches. It is shown that the signal/frame utilization is important to have fully utilized frames and not to waste any bandwidth. Nevertheless, putting signals with large periods into frames with small periods has a negative effect on the bandwidth utilization. That is, we conclude that Signal/Frame utilization should not be used directly to solve the frame packing problem.

In summary, our contributions to the frame packing problem on CAN and CAN-FD are as follows. As the first contribution, we developed two new frame packing approaches denoted as PBH and APBH. As the second contribution, we implemented all existing frame packing approaches as well as our new approaches in the form of a software tool. As the third contribution, we perform extensive experiments with systematically generated signal sets in order to evaluate the performance of the different approaches. As the main result, we show that our new algorithms perform best for all different types of signal sets on both CAN and CAN-FD.

# REFERENCES


[1] White paper: Can with flexible data-rate. 2011. http://www.bosch-semiconductors.de/media/pdf_1/canliteratur/can_fd.pdf.

[2] Specification version 1.0: Can with flexible data-rate. 2012. http://www.bosch-semiconductors.de/media/pdf_1/canliteratur/can_fd_spec.pdf.

[3] N. Audsley. On priority assignment in fixed priority scheduling. *Information Processing Letters*, 79(1):39–44, 2001.

[4] U. D. Bordoloi, S. Samii, and G. Motors. The frame packing problem for can-fd.

[5] C. Braun, L. Havet, N. Navet, et al. Netcarbench: a benchmark for techniques and tools used in the design of automotive communication systems. In *7th IFAC International Conference on Fieldbuses & Networks in Industrial & Embedded Systems-FeT'2007*, pages 321–328, 2007.

[6] M. Broy, I. H. Kruger, A. Pretschner, and C. Salzmann. Engineering automotive software. *Proceedings of the IEEE*, 95(2):356–373, 2007.

[7] E. G. Coffman Jr, M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: A survey. In *Approximation algorithms for NP-hard problems*, pages 46–93. PWS Publishing Co., 1996.

[8] S. C. C. A. R. Considerations. Sae j2056/1. *SAE Handbook*, pages 23–366, 1993.

[9] S. Corrigan. Introduction to the controller area network (can). *Application Report, Texas Instruments*, 2002.

[10] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien. Controller area network (can) schedulability analysis: Refuted, revisited and revised. *Real-Time Systems*, 35(3):239–272, 2007.

[11] F. Hartwich. Can with flexible data-rate. In *13th International CAN Conference (iCC2012), Hambach, Germany*, 2012.

[12] R. S. Marques, N. Navet, F. Simonot-Lion, et al. Frame packing under real-time constraints. In *5th IFAC International Conference on Fieldbus Systems and their Applications-FeT'2003*, pages 185–192, 2003.

[13] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert. Trends in automotive communication systems. *Proceedings of the IEEE*, 93(6):1204–1223, 2005.

[14] F. Polzlbauer, I. Bate, and E. Brenner. Optimized frame packing for embedded systems. *Embedded Systems Letters, IEEE*, 4(3):65–68, 2012.

[15] R. Saket and N. Navet. Frame packing algorithms for automotive applications. *Journal of Embedded Computing*, 2(1):93–102, 2006.

[16] K. Sandstrom, C. Norstom, and M. Ahlmark. Frame packing in real-time communication. In *Real-Time Computing Systems and Applications, 2000. Proceedings. Seventh International Conference on*, pages 399–403. IEEE, 2000.

[17] I. Standard. Iso 11898, 1993. *Road vehicles–interchange of digital information–Controller Area Network (CAN) for high-speed communication*, 1993.

[18] K. Tindell and A. Burns. Guaranteeing message latencies on control area network (can). In *Proceedings of the 1st International CAN Conference*, 1994.

[19] K. Tindell, A. Burns, and A. J. Wellings. Calculating controller area network (can) message response times. *Control Engineering Practice*, 3(8):1163–1169, 1995.

[20] K. Tindell and J. Clark. Holistic schedulability analysis for distributed hard real-time systems. *Microprocessing and microprogramming*, 40(2):117–134, 1994.

[21] K. Tindell, H. Hansson, and A. J. Wellings. Analysing real-time communications: controller area network (can). In *Real-Time Systems Symposium, 1994., Proceedings.*, pages 259–263. IEEE, 1994.