情報処理学会論文誌 Vol. 52 No. 7 2245–2255 (July 2011)

CANメッセージのオフセット決定手法

陳 暘[†1] 倉地 亮[†1]
曾 剛[†2] 高田広章[†1]

CAN（Controller Area Network）は，自動車の制御系ネットワークとして広く使われているネットワーク規格である．近年，自動車の制御系ネットワークを流れるデータ量が急増しており，CANの帯域を効率的に使用するために，CAN上でのメッセージのスケジューリング手法が重要となっている．CAN上でのメッセージ送信タイミングに適切なオフセットを付けることで，メッセージのスケジュール可能性を高くできることが知られている．本論文では，CAN上のメッセージに対して適切なオフセットを付けるための手法を提案する．また，提案した手法の有効性を確認するために，実際の車載システムで使用されているメッセージセットなどを用いた評価を行い，従来のオフセット決定手法よりも優れていることを示す．

An Offset Assignment Method for Messages of CAN

Yang Chen,[†1] Ryo Kurachi,[†1] Gang Zeng[†2]
and Hiroaki Takada[†1]

The Controller Area Network (CAN) as a network standard is widely employed in control network of vehicle. ln the last few years, the amount of data has been increasing rapidly in automotive control system network. With the purpose of improving CAN broadband efficiency, scheduling is considered to be particularly important. As we know, an appropriate offset assignment method for CAN messages could enhance schedulability. In this study, we propose a new method that provide a suitable offset for every message in CAN system. And we demonstrated its effects by using message sets of real vehicle network, etc. Compare with previous methods, the new method reveals its remarkable priority.

1. はじめに

近年，自動車制御システムの大規模化，複雑化にともなって，車載ネットワークにつながるECU（Electronic Control Unit）の数やシステム全体のメッセージ数は増加している．車載ネットワークはボデー系，制御系，マルチメディア系などに分類され，その中で特に高いリアルタイム性が重視される制御系のネットワークでController Area Network（CAN）[1]が広く使われている．

ECU数やシステム全体のメッセージ数が増加することで，ネットワーク上のバスの負荷率が上昇すると，メッセージの最大遅れ時間が長くなり，システムのリアルタイム性を確保することが難しくなってきている．ここでバスの負荷率とは，単位時間あたりに通信データがバスを占有する時間の割合のことである．また最大遅れ時間とは，メッセージの送信要求が行われてから，送信が完了するまでにかかる最大の時間のことである．周期的に送信されるCANメッセージに対して，オフセットと呼ばれる初回の送信要求時刻を適切に設定することができれば，同じ条件（バスの負荷率）の場合でも，最大遅れ時間を大幅に減少させることが可能である[2)–4)]．

本研究では，CANシステム全体のメッセージが時間制約を満たし，最大遅れ時間をできるだけ小さくできるようなオフセット決定手法を提案する．具体的には，1つのECU内において，他のECUが送信するメッセージを連続して妨げる可能性が小さくなるようオフセットの決定を行い，全体としてメッセージの配置が最良に近づくようにする．また，実際の自動車内ネットワークにおいて検討されているメッセージセットなどに対して提案手法を適用し，従来手法と比較することで，提案手法の有効性を確認する．

本論文では，まず2章で，本論文で解析する対象モデルについて述べる．3章では，提案手法の前提となるオフセット付きCANメッセージの最大遅れ時間の解析手法について説明する．4章で従来の研究について述べた後，5章で提案手法の詳細について説明する．6章では，提案手法をいくつかのメッセージセットに適用した例を説明し，提案手法の有効性を示す．

†1 名古屋大学大学院情報科学研究科
Graduate School of Information Engineering, Nagoya University
†2 名古屋大学大学院工学研究科
Graduate School of Engineering, Nagoya University

2. 解析モデル

本章では，本論文で使用する解析モデルについて説明する．

2.1 CAN（Controller Area Network）

CAN（Controller Area Network）は，1989 年にドイツの Robert Bosch 社により提案され規格化された通信プロトコルである．CAN の仕様では，OSI の参照モデルの物理層とデータリンク層のみが規定されており，これらの層はハードウェア（CAN コントローラ）で実現されている．CAN は最高で 1 Mbps の転送速度を実現できるブロードキャストバスである．CAN バス上では，最大長が与えられたメッセージ単位でデータがやりとりされる．CAN の仕様では，データフレーム，リモートフレーム，エラーフレーム，オーバーロードフレームの 4 種類のメッセージフレームが定義されている．ノード間でデータを通信するために用いられるデータフレームには，**図 1** に示すように，47 ビット分のプロトコル制御用の情報（ID，CRC，ACK や同期ビットなど）に加えて，0 から 8 バイトまでのデータ（ペイロード）を含めることができる．

メッセージには 11 ビットの ID（拡張仕様では 29 ビット）が割り付けられる．ID はメッセージの優先度を表し，受信時には，必要なメッセージだけを受信するためのフィルタリングにも使用される．ID がメッセージの優先度として用いられることが CAN での通信のリアルタイム性保証に関して最も重要となる．CAN のメッセージは ID の数値が小さいほど高い優先度を持つ．同時に 2 つ以上のノードが送信を開始しメッセージの衝突が起こった場合，衝突はワイヤード AND 機構による非破壊ビットワイズアービトレーション（Non-destructive bitwise arbitration）に従って解決される．具体的には，バス上の信号に優劣を設け，劣勢な信号（Recessive，論理 1）を優勢な信号（Dominant，論理 0）で上書きすることにより，劣勢な信号を送信したノードはその状態を検知し自身の送信を中断する．これにより，優先度の高いメッセージがバス上に確実に送り出されることになる．このようなアービトレーションを正しく行うために，同じ ID を持つ 2 つ以上のメッセージが同時に送信されないように，メッセージにはユニークな ID が割り当てらなければならない．

なお，CAN システムの構成によっては，高優先度メッセージの送信が，同じ ECU 内の低優先度メッセージに待たされる ECU 内優先度逆転が発生する場合がある[2]．本論文では，送信するメッセージの種類ごとにメールボックスを用意するなどの方法で，ECU 内優先度逆転が発生しないようになっているものと仮定する．

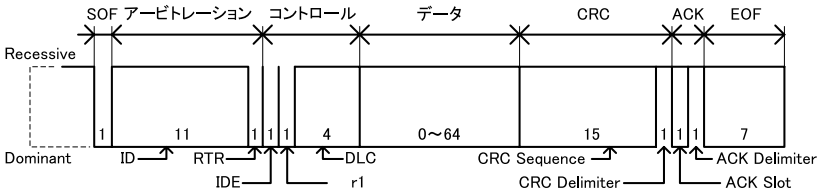


図 1　CAN のメッセージフォーマット
Fig. 1　CAN message format.

2.2 対象とするシステムの構成

本論文で扱う CAN システムは，N 個の ECU（$U_1, U_2, ..., U_N$）で構成され，各 ECU からは複数のメッセージが送信される．各 ECU は独自のタイマを用いて，各メッセージを周期的に送信要求する．メッセージの初回の送信要求は，ECU のタイマがメッセージごとに定められた値（これを，メッセージのオフセットと呼ぶ）になったときに行われ，以降は与えられた送信周期に従い，繰り返し送信要求が行われる．なお，各 ECU のタイマは ECU ごとに起動時に初期化され，各 ECU のタイマを同期させる機構はないため，異なる ECU からのメッセージの送信タイミングは同期していない．

以降では，メッセージ τ_i の優先度を P_i，最大送信時間を E_i，送信周期を C_i，オフセットを O_i と書くものとし，τ_i の情報を 4 つ組 (P_i, E_i, C_i, O_i) で表記する．優先度 P_i は，値が小さい方が，優先度が高いものとする．具体例として，U_1 の $\tau_i(3, 1, 6, 2)$ は**図 2** のように図示できる．

また，メッセージ τ_i が送信要求されてから，CAN バス上への送出が完了するまでの時間を，メッセージの遅れ時間と呼ぶ．時間制約を守るために重要となる指標は，遅れ時間の最大値（これを最大遅れ時間と呼び，R_i と書く）であり，遅れ時間として許容される最大時間を，メッセージのデッドライン D_i と呼ぶ．すなわち，メッセージ τ_i は，$R_i \leq D_i$ のと

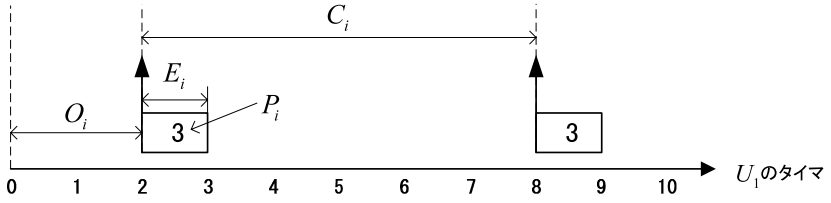


図 2　CAN メッセージの表記
Fig. 2　Notation of CAN messages.

きに時間制約を満たすものとする．

なお，本論文では，メッセージの最大送信時間 E_i に加えて，送信周期 C_i とオフセット O_i も，1 ビットの送信時間 τ_{bit} の整数倍で表現できるものとする．これは，メッセージの送信周期とオフセットは通常ミリ秒単位であり，τ_{bit} と比べてきわめて大きい値であるため（たとえば，500 kbps の CAN バスの τ_{bit} は 2 μs である），τ_{bit} の整数倍であると近似しても支障がないためである．

また，実際の車載ネットワークでは，送信要求が突発的に発生する非周期メッセージも使用されるが，非周期メッセージに対してはオフセットが意味を持たないため，本論文では考慮しないこととする．

2.3　メッセージに対する時間制約

CAN メッセージに対する時間制約は，CAN メッセージに載せるデータが生成されてから，そのメッセージが送信要求され，CAN バス上への送出が完了するまでの全体の時間に対して課されると考えることができる．

前節で述べたとおり，本論文では，メッセージは周期的に送信要求されるものとしているが，多くの車載システムにおいて，CAN メッセージに載せるデータが生成されるタイミングと，メッセージが送信要求されるタイミングは，同期しているとは限らない．これは，次のような理由による．

- CAN メッセージに載せるデータは，周期的に実行される ECU 内の制御演算により生成されるのが一般的であるが，制御演算にかかる時間は周期ごとに変化するために，データの生成タイミングは周期ごとに一定しない．
- 1 つの CAN メッセージには，複数のデータが載せられるが，それらのデータが同時に生成されるとは限らない．
- ECU 内の制御演算の実行周期と，CAN メッセージの送信周期は一致するとは限らない．前者の方が短い場合が多い．

CAN メッセージに載せるデータの生成タイミングとメッセージの送信要求タイミングが同期しないことから，データが生成されてからメッセージが送信要求されるまでの時間が最大になるのは，データが生成される直前に，メッセージが送信要求されていた場合である．この場合，生成されたデータを載せたメッセージが送信されるのは，1 送信周期後となる．そのため，データの生成からメッセージの送信要求までの最大時間は，メッセージの送信周期と一致する．

以上のことから，CAN メッセージに載せるデータが生成されてからメッセージの送出が

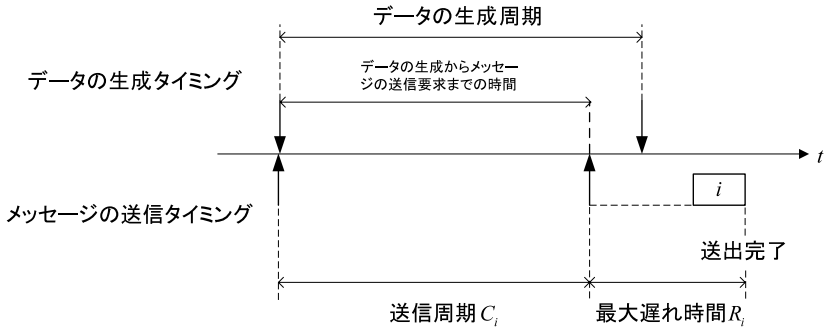


図 3　データの生成からメッセージの送信完了までの最悪状況
Fig. 3　The worst case from data generation to message transmission.

完了するまでの最大時間は，メッセージの送信周期 C_i に最大遅れ時間 R_i を加えた時間となる（**図 3**）．そのため，データの生成からメッセージの送出完了までにかかる時間として許容される最大時間を，メッセージの送信周期 C_i とメッセージのデッドライン D_i に分配することで，送信周期とデッドラインを決めることになる．このことから，本論文では，メッセージのデッドラインは，メッセージの送信周期にデッドライン率と呼ぶ定数値を掛けた値であるものとする．たとえば，デッドライン率が 30%の場合には，送信周期 10 のメッセージのデッドラインは 3 である．実際の車載システムの設計においても，個々のメッセージのデッドラインが個別に決定されているわけではなく，デッドライン率を目安に時間制約を満たしているかどうかを判断している．

また，この考え方にあわせて，本論文では，メッセージの送信がどの程度遅れるかの評価指標として，次の式で表される最大遅延率 L_i を用いる．

$$L_i = (R_i/C_i) * 100\% \tag{1}$$

たとえば，送信周期 10 のメッセージの最大遅れ時間が 2 の場合，最大遅延率は 20%である．最大遅延率がデッドライン率以下の場合，メッセージは時間制約を満たしていることになる．

2.4　オフセットの効果

CAN メッセージに適切にオフセットを与えることで，次の 2 つの効果がある．同じ ECU 内のメッセージにとっては，ECU 内の複数のメッセージの送信要求が同時に発生することを回避でき，メッセージの最大遅れ時間を短縮することができる．また，他の ECU の低優先度メッセージにとっては，最悪状況で連続して送信される高優先度メッセージが少なくな

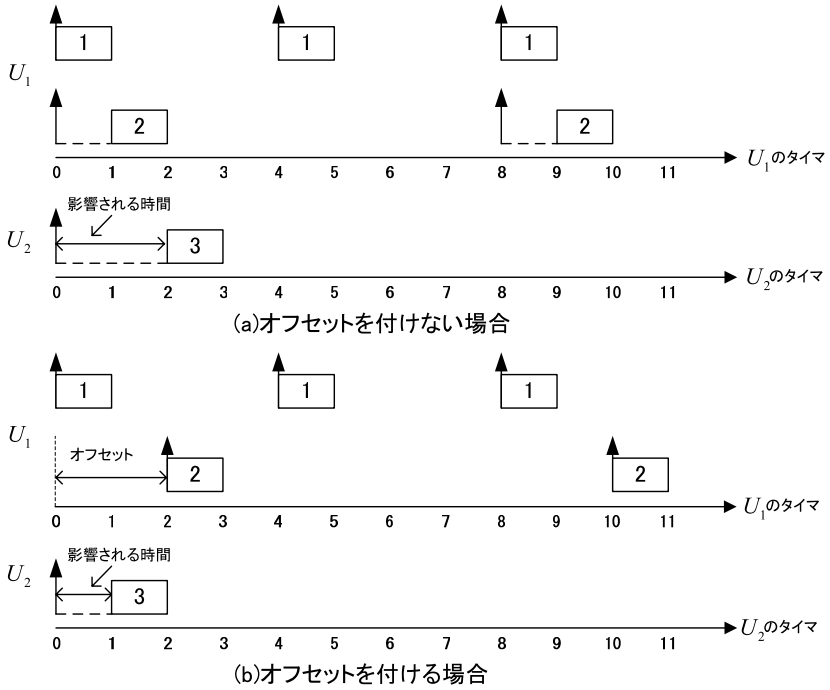


図 4　オフセットの効果
Fig. 4　Effects of offset.

るため，その低優先度のメッセージの最大遅れ時間も短縮することができる．

図 4 は，U_1 のメッセージ $\tau_1(1,1,4,0)$，$\tau_2(2,1,8,0)$ と，U_2 のメッセージ $\tau_3(3,1,16,0)$ を用い，オフセットを付けた場合と付けない場合の各メッセージの送信状況の例を図示したものである．この図の例は，U_1 のタイマが 0 のときに，U_2 のタイマも 0 であった場合を示している．オフセットを付けない場合（いい換えると，すべてのメッセージのオフセットが 0 の場合）には，τ_2 は τ_1 の送信が完了するまで待たされ，τ_3 は τ_2 と τ_1 の送信が完了するまで待たされることになる．それに対して，U_1 において，τ_1 にオフセット 0，τ_2 にオフセット 2 を付与すると，U_1 内で 2 つのメッセージが衝突することがなくなるため，τ_2 が τ_1 の送信完了を待つことがなくなり，さらに τ_3 を妨げる最大時間も小さくなる．ここで，各 ECU は非同期に動作しており，U_2 が送信するメッセージは τ_3 しかないため，τ_3 にオフセットを付与しても意味がない．

3. 最大遅れ時間の解析手法

メッセージが時間制約を満たすか判断するには，最大遅れ時間を求めることが必要である．そこで本章では，オフセット付きメッセージに対する最大遅れ時間の解析手法について述べる．

本論文で使用する最大遅れ時間解析手法は，文献 2) で提案されたオフセット付き CAN メッセージの最大遅れ時間解析手法に対して，ECU 内優先度逆転の効果を考慮した部分を削除し，文献 5)，6) で指摘されたオフセットのない CAN メッセージの最大遅れ時間解析手法の間違いを考慮した修正を加えたものである．

具体的には，U_I のメッセージ τ_i の最大遅れ時間は，τ_i 自身の送信時間に，U_I（自 ECU）の高優先度メッセージが τ_i を妨げる最大時間と，U_I 以外の ECU（他 ECU）の高優先度メッセージが τ_i を妨げる最大時間を加算して計算する．本章では，他 ECU と自 ECU の高優先度メッセージが，メッセージ τ_i を妨げる最大時間を求める方法を順に説明し，それを用いて τ_i の最大遅れ時間を計算する方法について述べる．

3.1 他 ECU の高優先度メッセージの影響

他 ECU の高優先度メッセージが，メッセージ τ_i を妨げる最大時間を，Maximum Interference Function（以下 MIF と略記）で表現する．MIF とは，マルチフレームタスクモデル[7] において，あるタスクが時間 t の間に，それより低優先度のタスクの実行を妨げる最大時間を与える関数である[8),9)]．

CAN システムにおいて，ある ECU が送信するメッセージの集合を，それらのメッセージの送信周期の最小公倍数（LCM）を周期とするマルチフレームタスクと見なすことにより，MIF の手法を適用することができる．すなわち，他 ECUU_J が，メッセージ τ_i を妨げる最大時間は，U_J が送信する τ_i より優先度が高いメッセージの MIF $M_J[P_i](t)$ から計算することができる．ここで，$M_J[P_i](t)$ は次の式で与えられる．

$$M_J[P_i](t) = \max_{1 \le k \le n'_J} F_J^k[P_i](t)$$
$$n'_J = \sum_{\tau_j \in U_J \cap \tau_j \in hp(P_i)} LCM_{\{\tau_j\}}/C_j \qquad (2)$$

ここで $hp(P_i)$ は τ_i より優先度が高いメッセージであり，$LCM_{\{\tau_j\}}$ は U_J に含まれる τ_i より優先度が高いメッセージの送信周期の LCM である．また，n'_J は，U_J が $[0, LCM_{\{\tau_j\}}]$

の間に送信するメッセージの総数である．$F_J^k[P_i](t)$ は，U_J が k 番目のメッセージを送信要求する時刻から開始して，U_J が送信する τ_i より高優先度のメッセージが τ_i を妨げる時間を与える関数 Interference Function（以下 IF と略記）である．

ここで具体的な例を用いて MIF の計算方法を説明する（**図 5**）．$\tau_1(1,1,8,0)$，$\tau_2(2,1,8,3)$，$\tau_3(3,1,8,4)$ は，U_J が送信する τ_i より高優先度のメッセージであるとする．U_J が送信する τ_i より高優先度のメッセージの MIF を求めるために，τ_1，τ_2，τ_3 のそれぞれの送信要求時刻から LCM までの IF を求める．それらの IF の最大値の関数が U_J の MIF である．この例から求めた IF と MIF を，図 5 で示す．

ECU ごとに τ_i より高優先度のメッセージの MIF が求まると，それらの MIF をすべて飽和加算することで，τ_i が他 ECU の高優先度メッセージにより妨げられる最大時間を求めることができる．ここで MIF の飽和加算とは，複数の MIF を，最大の傾きを 1 に制限して加算する演算である．例として，τ_i より高優先度のメッセージとして，U_J が送信するメッセージ $\tau_1(1,1,8,0)$，$\tau_2(2,1,8,3)$，$\tau_3(3,1,8,4)$ と，U_K が送信するメッセージ $\tau_4(4,1,4,0)$ がある場合を考える．このとき，U_J のメッセージの MIF $M_J[P_i](t)$ と，U_K のメッセージの MIF $M_K[P_i](t)$ を飽和加算した関数 $M_J[P_i](t) \oplus M_K[P_i](t)$ は，**図 6** のようになる．τ_i が送信できるのは，飽和加算で得られた MIF の傾きが最初に 0 になる時刻 3 であり，τ_i が高優先度メッセージにより妨げられる最大時間が 3 ということになる．

3.2　自 ECU の高優先度メッセージの影響

自 ECU が送信する τ_i より高優先度のメッセージで，τ_i より先に送信要求されたものの送信が遅れた場合，τ_i に影響を与える可能性がある．また，τ_i の送信が遅れた場合，τ_i より後に送信要求される高優先度メッセージに追い越される場合がある．さらに，τ_i の送信が遅れることにともなって，τ_i の次の周期の送信に影響を与える可能性も存在する[5),6)]．

これらを考慮し，U_I が送信するメッセージ τ_i の最大遅れ時間は，次の方法で求めることができる．

(1)　前節の方法に従って，U_I 以外の各 ECU ごとに τ_i より高優先度のメッセージの MIF を求め，それらの MIF をすべて飽和加算した MIF を求める．

(2)　高優先度メッセージが送信要求されたときに，すでに低優先度メッセージの送信が開始されていると，1 メッセージ分の優先度逆転が生じる[⋆1]．この効果を考慮するために，上で求めた MIF に，U_I 以外の ECU が送信する低優先度メッセージの中で，送

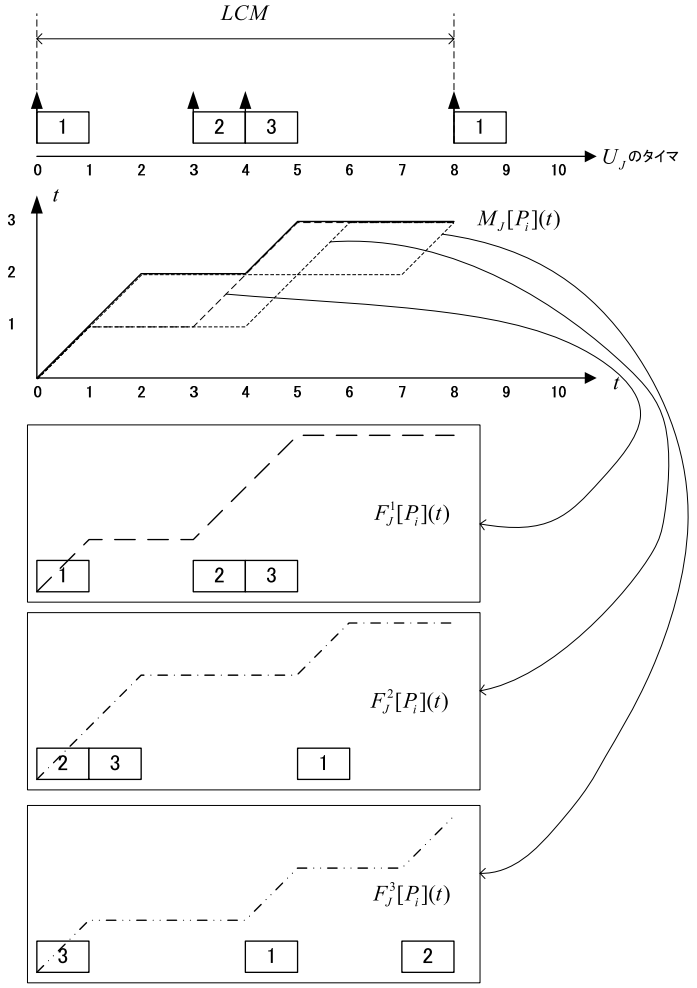


図 5　CAN メッセージの MIF の例
Fig. 5　MIF of CAN messages.

⋆1 この優先度逆転は，バス上で発生するもので，システム構成によらず生じる．

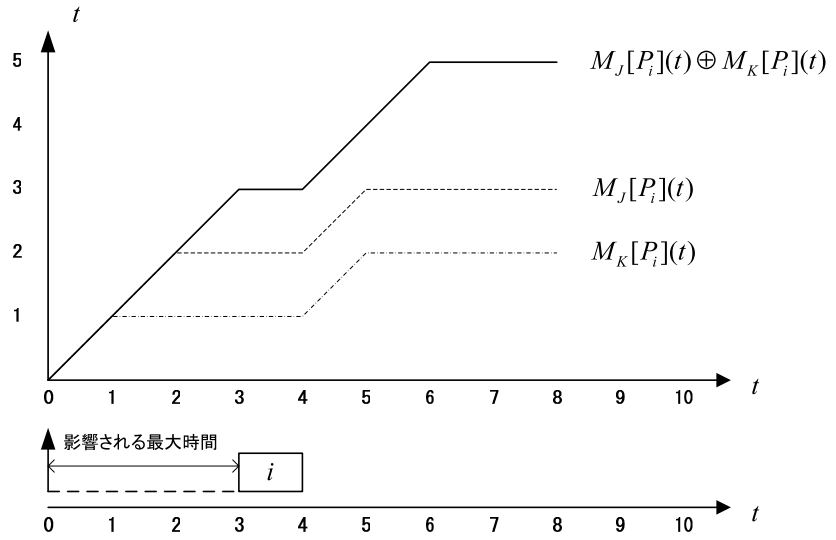


図 6　MIF の飽和加算の例
Fig. 6　Saturation addition of MIF.

信時間が最長となるメッセージの送信時間を飽和加算する．

(3)　τ_i の送信要求時刻を開始時刻として，U_I が送信する τ_i より高優先度メッセージの IF を求め，その IF を上で求めた MIF に飽和加算し，総 MIF を求める．総 MIF の傾きが最初に 0 になる時刻が，τ_i を送信開始できる時刻である．τ_i の送信要求時刻からその時刻までの時間に，τ_i の送信時間 E_j を加えたものが，最大遅れ時間の候補の 1 つになる．

(4)　U_I が送信する τ_i より高優先度または同じ優先度のメッセージで，τ_i より先に送信要求されるメッセージ τ_j を探す．τ_j の送信要求時刻を開始時刻として，U_I が送信する τ_i より高優先度メッセージの IF を求め，その IF を上で求めた MIF に飽和加算し，総 MIF を求める．総 MIF の傾きが最初に 0 になる時刻が，τ_i を送信開始できる時刻である．τ_i の送信要求時刻からその時刻までの時間に，τ_i の送信時間 E_j を加えたものが，最大遅れ時間の候補の 1 つになる．

(5)　上記の計算を，τ_i に先行するすべての高優先度または同じ優先度のメッセージ τ_j に対して繰り返し行い，導出した最大遅れ時間の候補の中の最大値が，τ_i の最大遅れ時間である．

4. 従来の研究

松谷らは，各 ECU に対して，その ECU が送信するメッセージ間の送信衝突を回避するようにオフセットを付ける手法を提案した[3]．この手法は，ECU 間のメッセージの送信衝突を考慮していないため，バスの負荷率が高いシステムにとってすべてのメッセージの時間制約を満たせるオフセットを決定することは難しい．

Grenier らは，各 ECU に対して，その ECU が送信するメッセージの送信要求時刻を分散するようにオフセットを付ける手法を提案している[4]．送信要求時刻を分散することで，メッセージの送信要求時刻の間隔が長くなり，ECU 間のメッセージの送信衝突を削減できる．Grenier らが提案する手法において，U_I が送信するメッセージのオフセットを決定する手順は次のとおりである．

(1)　U_I 内のオフセットを決定していないメッセージの中で，最も周期の短いメッセージ τ_i に対して，$[0, C_i)$ の間で，送信要求されるメッセージがない最長の区間 (t_0, t_1) を探す．

(2)　τ_i のオフセット O_i を，$(t_0 + t_1)/2$ に設定する．

(3)　(1) と (2) を，すべてのメッセージのオフセットを決定するまで繰り返す．

最大遅れ時間の解析手法から分かるように，あるメッセージ τ_i の最大遅れ時間は，P_i 以上の優先度を持つメッセージの MIF により決定される．Grenier らの手法は，ECU 間のメッセージの送信衝突をある程度削減できるが，MIF が最小になるとは限らず，メッセージの最大遅れ時間を最小に抑えることができない．

5. 提 案 手 法

あるメッセージの最大遅れ時間を削減するには，そのメッセージより高い優先度のメッセージの MIF を削減するようにオフセットを付けるのが有効である．つまり，各メッセージに対して，システム内のより高優先度のメッセージの MIF を最小にするようにオフセットを付けることができれば，メッセージの最大遅れ時間を最小に抑えることができる．しかし実際のシステムでは，あるメッセージに着目して最適なオフセットを付けた場合，他のメッセージにとっては必ずしも最適なオフセットになるとは限らない．

そこでまず，ECU ごとに，その ECU が送信するすべてのメッセージの MIF を最小にするようにオフセットを付ける．次に，システムのすべてのメッセージの最大遅れ時間を求め，

時間制約を満たさないメッセージに着目して，オフセットを再調整するアプローチをとる．

ここで，MIF を最小にするといった場合に，どの時刻における MIF を最小にするかが問題になる．着目するメッセージが定まっていれば，そのメッセージのデッドラインにおける MIF を最小にすればよいが，ここでは着目するメッセージを 1 つに定められないことから，MIF を時刻 0 から LCM まで積分した値（Integration of Maximum Interference Function，以下 IMIF と略記）を最小にする．U_I が送信するすべてのメッセージの MIF を $M_I(t)$，IMIF を I_I と表記すると，I_I は次の式で求めることができる．

$$I_I = \int_0^{LCM_I} M_I(t)dt \tag{3}$$

5.1　ECU ごとのオフセット決定

各 ECU の IMIF を最小にするオフセットの決定は，最適化アルゴリズムの 1 つである Simulated Annealing（焼きなまし法，以下 SA と略記）を用いて行う[10]．各 ECU U_I のオフセットを探索するアルゴリズムは，次のとおりである．

(1)　温度パラメータ T の初期値とアルゴリズムの探索回数上限を指定する．
(2)　U_I のすべてのメッセージ $\{\tau_i\}$ に対して，オフセット $\{O_i\}$ の初期値をランダムに選び，I_I を計算する．その $\{O_i\}$ と I_i を “現行解” および “最良解” として記録する．
(3)　U_I のメッセージの中からランダムに τ_j を選び，そのオフセット O_j をプラス 1 あるいはマイナス 1 に変更し，I_I を再計算する．変更後の $\{O_i\}$ と I_I は “新解” として記入し，“新解” と “現行解” の IMIF の差 Δ を求める．
(4)　“新解” の IMIF が “最良解” の IMIF よりも小さい場合，“新解” を “最良解” として記録する．
(5)　[0, 1) の範囲で乱数 R を生成する．$R < \Delta/T$ の場合，“新解” を “現行解” とする．
(6)　温度パラメータ T を減らし，探索回数をインクリメントする．探索回数が探索回数上限を超えない場合，(3) に戻って探索を続ける．
(7)　探索回数上限を超えると，探索を終了し，“最良解” に対する $\{O_i\}$ を，U_I のメッセージのオフセットとする．

5.2　オフセットの再探索の必要性

5.1 節のアルゴリズムによりオフセットを決定したら，3 章のアルゴリズムにより各メッセージの最大遅れ時間を求める．その結果，時間制約を満たさないメッセージがある場合には，そのメッセージより優先度が高いメッセージを含む ECU に対して，オフセットの再探索を行う．

表 1　異なるオフセットを持つメッセージの最大遅れ時間の比較
Table 1　Comparison of worst case response time in messages with different offsets.

U_1					
τ_i	P_i	E_i	C_i	O_i^1	O_i^2
τ_1	1	3	8	0	0
τ_2	2	2	8	3	4
τ_4	4	1	8	6	3
U_2					
τ_i	P_i	E_i	C_i	$R_i^1(L_i^1)$	$R_i^2(L_i^2)$
τ_3	3	1	8	6(75%)	4(50%)

再探索の必要性を**表 1** のメッセージセット例を用いて説明する．ここでは，τ_3 を対象メッセージとし，デッドライン率が 60%と仮定する．U_1 の 3 つのメッセージのオフセットを，表 1 の O_i^1 のように設定すると，I_1 は最小になる．このとき，τ_3 の最大遅れ時間（R_i^1）は 6 であり，最大遅延率（L_i^1）は 75%であるため，時間制約を満たさない．ここで，τ_3 よりも低い優先度のメッセージ τ_4 の影響を含む I_1 を用いて O_i^1 を決定したが，τ_3 の最大遅れ時間を小さくするためには，τ_4 は考慮しない方がよい．

そこでまず，U_1 の τ_3 より高い優先度のメッセージ τ_1 と τ_2 の IMIF を小さくするようにオフセットを決め，その後，τ_3 より低い優先度のメッセージ τ_4 のオフセットを決めると時間制約を満たせることがある．たとえば表 1 の O_i^2 のように設定すると，τ_3 の最大遅れ時間（R_i^2）は 4 になり，最大遅延率（L_i^2）は 50%になり，時間制約を満たせる．O_i^1 と O_i^2 に対する U_1 のすべてのメッセージの MIF と，τ_3 より高い優先度メッセージの MIF を**図 7** に示す．このように，時間制約を満たさないメッセージがある場合には，時間制約を満たさなかったメッセージを優先してオフセットを決定する方が，最大遅れ時間の観点から良い結果が得られることがある．

5.3　オフセットの再探索のアルゴリズム

時間制約を満たせなかったメッセージに重みを付けてオフセットを再探索するために，ECU ごとにオフセットを決定する際の評価関数を，式 (3) の I_I から式 (4) の I_I' に変更する．

$$\begin{aligned} I_I[P_i] &= \textstyle\int_0^{LCM_I} M_I[P_i](t)dt \\ I_I' &= I_I + \sum_i (W[P_i] * I_I[P_i]) \end{aligned} \tag{4}$$

ここで $M_I[P_i](t)$ と $I_I[P_i]$ はそれぞれ，U_I 内の P_i 以上の優先度を持つメッセージの MIF と IMIF である．$W[P_i]$ は，τ_i の重みを制御するためのパラメータであり，τ_i の重みパラ

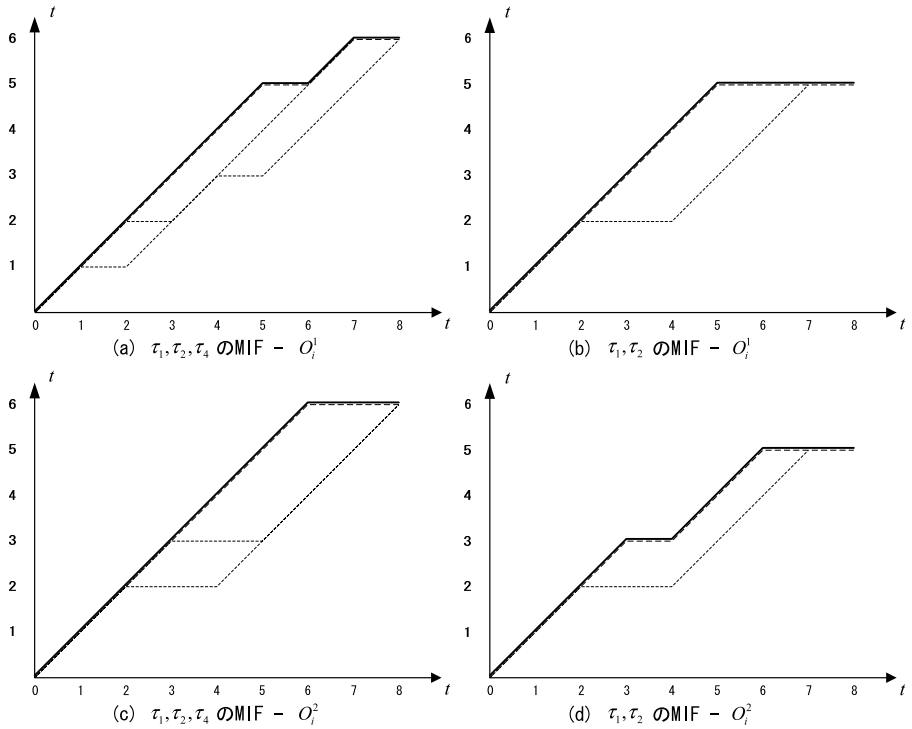


図 7　異なるオフセットを持つメッセージの MIF の比較
Fig. 7　Comparison of MIF in messages with different offsets.

メータ（Weighting Factor，以下 WF と略記）と呼ぶ．

複数のメッセージが時間制約を満たせなかった場合には，次回の再探索はそれらのメッセージの中で最も最大遅延率が大きいメッセージの重みを増加させて行う．WF の制御および再探索は，以下の手順で行う．

(1)　すべての WF を 0 に初期化する．
(2)　各 ECU に対して I_i' を最小にするようなオフセットを探索する．
(3)　各メッセージの最大遅れ時間を計算する．時間制約を満たさないメッセージがある場合，それらのメッセージの中で最大遅延率が最も大きいメッセージ τ_k に対して，$W[P_k]$ を 1 増加させ，(2) の先頭に戻って再探索を行う．

ここで具体的な例を用い，再探索の方法を説明する．U_1 のメッセージ τ_1，τ_2，τ_5，τ_6 と U_2 のメッセージ τ_3，τ_4，τ_7，τ_8 は，システムが送信するすべてのメッセージであり，それらのメッセージの優先度は $P_1 < P_2 ... < P_8$ と設定している．最初に，すべてのメッセージに重みを付けない状態でオフセットを決定し，時間制約を満たせなかったメッセージは τ_4 と τ_7 であり，τ_4 の最大遅延率の方が大きかったものと仮定する．1 回目の再探索を行うとき，WF の設定は $W[P_4] = 1$ となり，各 ECU のオフセット探索のための評価関数は下の式になる．

$$I_I' = I_I + 1 * I_I[P_4]$$

この評価関数による再探索の結果，時間制約を満たせなかったメッセージは τ_4 と τ_7 であり，τ_4 の最大遅延率の方が大きい場合と仮定する．2 回目の再探索では，$W[P_4] = 2$ と WF の設定を変更する．すなわち，各 ECU のオフセット探索のための評価関数は下の式になる．

$$I_I' = I_I + 2 * I_I[P_4]$$

2 回目の再探索の結果，時間制約を満たせなかったメッセージが τ_7 のみであったとすると，3 回目の再探索では $W[P_7] = 1$ とし，各 ECU のオフセット探索のための評価関数は下の式になる．

$$I_I' = I_I + 2 * I_I[P_4] + 1 * I_I[P_7]$$

5.4　アルゴリズムのまとめ

以上の内容をまとめると，再探索を含むオフセット付けアルゴリズムは，以下のとおりとなる．

(1)　すべての WF を 0 に初期化し，プログラムの実行回数の上限を指定する．
(2)　ECU ごとに SA アルゴリズムを用いてすべてのメッセージのオフセットを決定し，各メッセージの最大遅れ時間と最大遅延率を求める．
(3)　時間制約を満たせなかったメッセージがある場合には，満たせなかったメッセージの中で最大遅延率が最も大きいメッセージに対して WF を増加させ，(2) に戻って再探索を行う．
(4)　すべてのメッセージが時間制約を満たした場合，または設定した実行回数の上限を超えた場合には，プログラムを終了する．

なお，以上で提案したオフセット決定手法では，ECU ごとに SA を用いてオフセットを決定しているが，全 ECU に対して SA を用いてオフセットを決定する方法も考えられる．しかし，この方法では，ECU が追加された場合や，ある ECU に送信するメッセージが追加された場合に，他の ECU が送信するメッセージのオフセットも変更しなければならず，

実際の車載システムへの適用が現実的でないため，採用しなかった．

6. 適　　用

本章では，提案したオフセット決定手法を，2 社の自動車メーカから提供を受けた車載ネットワークのメッセージセットと，文献 11) のツールを使用して作成したメッセージセットに対して適用し，提案手法の評価を行う．

本適用実験においては，まず，メッセージに重みを与えずにオフセットを決定する．この時点で，最大遅延率が最も大きいメッセージの最大遅延率（以下，最大遅延率の最大値という）をデッドライン率パラメータとおいた場合には，すべてのメッセージが時間制約を満たすことになる．次に，デッドライン率パラメータをより小さい値に設定し，提案手法によりオフセットを決定する．これを，すべてのメッセージの時間制約を満たせるオフセットが見つからなくなるまで，デッドライン率パラメータを徐々に小さくして繰り返す．ここでのデッドライン率パラメータとは，提案手法のアルゴリズムの効果を評価するために設定するデッドライン率を意味するパラメータである．

また，提案手法の有効性を示すために，Grenier らのアルゴリズム[4] と比較する．比較は，メッセージの平均最大遅延率 $\bar{L}_i$ と最大遅延率の最大値 $\max L_i$ により行う．

実験 1 では，自動車メーカから提供を受けたメッセージセット（バスの負荷率は 35%，ECU 数は 15 個，メッセージ数は 65 個）を用い，本手法の効果を検証する．**表 2** で示すとおり，文献 4) で提案した従来の手法と比べると，本手法でメッセージに重みを付けずに決定したオフセットは，メッセージの平均最大遅延率を 3.66%削減でき，最大遅延率の最大値を 7.67%削減できた．

ここで，最大遅延率が最大になったメッセージを時間制約を満たさないと扱うために，デッドライン率パラメータを 32.3%に設定し，提案した手法により再探索を行った．その結果，従来の手法と比べて，平均最大遅延率を 4.54%，最大遅延率の最大値を 8.24%削減でき，時間制約を満たさないメッセージがなくなった．

表 2　実験 1
Table 2　Experiment 1.

探索方法	$\bar{L}_i$（削減率）	$\max L_i$（削減率）	時間制約を満たさない数
従来の方法	8.33%(-)	35.20%(-)	1
本手法 1 回目	8.03%(3.66%)	32.50%(7.67%)	1
本手法 2 回目	7.95%(4.54%)	32.30%(8.24%)	0

次に実験 2 では，他の自動車メーカから提供を受けたメッセージセット（バスの負荷率は 65%，ECU 数は 14 個，メッセージ数は 68 個）を用い，実験を行った．**表 3** に示すとおり，本手法でメッセージに重みを付けずに決定したオフセットは，従来の手法と比べて，メッセージの平均最大遅延率を 4.91%，最大遅延率の最大値を 11.37%削減できた．また，デッドライン率パラメータを 32.5%と設定し，再探索を行った結果，従来の手法と比べて，メッセージの平均最大遅延率を 7.66%，最大遅延率の最大値を 14.86%削減できた．従来の手法では時間制約を満たさないメッセージが 1 つあるが，本手法の結果では時間制約を満たさないメッセージがない．

以上の実験は，各メッセージセット内の ECU 数が多く，各 ECU 内のメッセージ数は少なく，ECU ごとの負荷率が低いため，オフセットを付与することの有効性が低くなっていると考えられる．ECU ごとの負荷率が高い場合の本手法の有効性を検証するために，文献 11) のツールによりそのようなメッセージセットを作成して，実験を行った．

実験 3 のメッセージセットは，バスの負荷率は 55%，ECU 数は 5 個，メッセージ数は 75 個である．**表 4** で示すとおり，従来の手法と比べると，本手法でメッセージに重みを付けずに決定したオフセットは，メッセージの平均最大遅延率を 16.13%，最大遅延率の最大値を 3.29%削減できた．デッドライン率パラメータを 15.00%とした場合には，時間制約を満たさないメッセージ数は 5 から 2 まで削減できた．さらに，時間制約を満たさないメッセージの中で最大遅延メッセージに重みを付けて再探索すると，従来手法と比べて，平均最大遅

表 3　実験 2
Table 3　Experiment 2.

探索方法	$\bar{L}_i$（削減率）	$\max L_i$（削減率）	時間制約を満たさない数
従来の方法	18.95%(-)	38.13%(-)	1
本手法 1 回目	18.02%(4.91%)	33.79%(11.37%)	1
本手法 2 回目	17.49%(7.70%)	32.71%(14.21%)	1
本手法 3 回目	17.50%(7.66%)	32.46%(14.86%)	0

表 4　実験 3
Table 4　Experiment 3.

探索方法	$\bar{L}_i$（削減率）	$\max L_i$（削減率）	時間制約を満たさない数
従来の方法	6.59%(-)	16.40%(-)	5
本手法 1 回目	5.53%(16.13%)	15.86%(3.29%)	2
本手法 2 回目	5.50%(16.49%)	14.78%(9.88%)	0

表 5　実験 4
Table 5　Experiment 4.

探索方法	$\bar{L}_i$（削減率）	$\max L_i$（削減率）	時間制約を満たさない数
従来の方法	10.83%(-)	29.38%(-)	11
本手法 1 回目	8.04%(25.69%)	20.88%(28.93%)	5
本手法 2 回目	7.69%(29.01%)	19.80%(32.61%)	2
本手法 3 回目	7.27%(32.80%)	18.82%(35.94%)	0

延率を 16.49%，最大遅延率の最大値を 9.88%削減でき，時間制約を満たさないメッセージはなくなった．

実験 4 のメッセージセットは，バスの負荷率は 65%，ECU 数は 5 個，メッセージ数は 74 個である．**表 5** で示すとおり，従来の手法と比べると，本手法でメッセージに重みを付けずに決定したオフセットは，メッセージの平均最大遅延率を 25.69%，最大遅延率の最大値を 28.93%削減できた．デッドライン率パラメータを 19.00%とした場合には，時間制約を満たさないメッセージ数は 11 から 5 まで削減できた．さらに，時間制約を満たさないメッセージの中で最大遅延メッセージに重みを付けて再探索を繰り返すと，最終的に，従来手法と比べて，平均最大遅延率を 32.80%，最大遅延率の最大値を 35.94%削減でき，すべてのメッセージは時間制約を満たせた．

以上より，ECU ごとの負荷率が高い場合において，本手法の有効性が特に高いことが分かる．

7. 結　　論

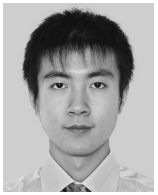
本論文では，車載ネットワークに CAN を使用する場合に，メッセージのオフセットを決定する手法を提案した．4 種類のメッセージセットを用いた評価の結果から，提案手法は，従来手法と比べて，メッセージの平均最大遅延率を約 4〜33%，最大遅延率の最大値を約 3〜35%を削減できた．バスの負荷率，特に ECU ごとの負荷率が高い場合，本手法の有効性が高い．この結果より，本手法を用いることで時間制約を満たすメッセージ数を増やすことができる．

本提案手法では，時間制約を満たさないメッセージより高い優先度のメッセージの MIF に対して重みを付けたが，MIF の積分区間をそのようなメッセージのデッドラインまでとする方法や，そのようなメッセージのデッドライン時点での MIF の値を評価関数とする方法も考えられる．このような方法の有効性評価に関しては，今後の課題である．

参 考 文 献

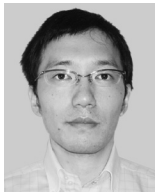
1) International Organization for Standardization, Road vehicles, Controller area network (CAN), Part1: Data link layer and physical signaling, ISO IS11898-1 (2003).
2) 飯山真一，冨山宏之，高田広章，城戸正利，細谷伊知郎：オフセット付き CAN メッセージの最大遅れ時間解析，情報処理学会論文誌：コンピューティングシステム，Vol.45, No.SIG_11(ACS_7), pp.455–464 (2004).
3) 松谷元気，飯山真一，冨山宏之，高田広章：CAN におけるメッセージスケジューリング手法，情報処理学会研究報告（2004 年実時間処理に関するワークショップ RTP2004），Vol.2004, No.33, pp.81–86 (2004).
4) Grenier, M., Havet, L. and Navet, N.: Pushing the limits of CAN – Scheduling frames with offsets provides a major performance boost, *Proc. 4th European Congress Embedded Real Time Software* (*ERTS 2008*), Toulouse, France (2008).
5) Bril, R.J., Lukkien, J.J., Davis, R.I. and Burns, A.: Message response time analysis for ideal controller area network (CAN) refuted, *Proc. 5th International Workshop on Real-Time Networks* (*RTN'06*) (July 2006).
6) Davis, R.I., Burns, A., Bril, R.J. and Lukkien, J.J.: Controller Area Network (CAN) Schedulability Analysis: Refuted, Revisited and Revised, *Real-Time Systems*, Vol.35, No.3, pp.239–272 (2007).
7) Mok, A.K. and Chen, D.: A multiframe model for real-time tasks, *Proc. Real-Time Systems Symposium*, pp.22–29 (1996).
8) 高田広章，坂村　健：マルチフレームタスクセットの静的優先度スケジューリングによるスケジュール可能性，信学技報（1997 年実時間処理に関するワークショップ RTP'97），Vol.96, No.596, pp.29–35 (1997).
9) Takada, H. and Sakamura, K.: Schedulability of generalized multiframe task sets under static priority assignment, *Proc. Real-Time Computing Systems and Applications*, pp.80–86 (1997).
10) Kirkpatrick, S., Gelatt Jr, C. and Vecchi, M.: Optimization by simulated annealing, *Science*, Vol.220, No.4598, pp.671–680 (1983).
11) Braun, C., Havet, L. and Navet, N.: NETCARBENCH: A benchmark for techniques and tools used in the design of automotive communication systems, *Proc. 7th IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems* (*FeT 2007*) (Nov. 2007). Software and manual available at http://www.loria.fr/navet/netcarbench/

(平成 22 年 10 月 29 日受付)
(平成 23 年　4 月　8 日採録)



陳　　暘

2007 年中国西安交通大学大学院情報工学専攻修士課程修了．現在，名古屋大学大学院・情報システム学専攻博士後期課程に在学．車載ネットワークのスケジューリングおよび最大遅れ時間の解析に関する研究に従事．修士（工学）．



倉地　　亮（正会員）

名古屋大学大学院情報科学研究科附属組込みシステム研究センター研究員．2000 年東京理科大学基礎工学部電子応用工学科卒業後，アイシン・エィ・ダブリュ株式会社にてカーナビゲーションシステムの開発に従事．2007 年東京理科大学専門職大学院総合科学技術経営研究科技術経営専攻修了後，現職．車載 LAN に関する研究に従事．

曾　　剛

2006 年千葉大学大学院自然科学研究科博士課程修了．同年名古屋大学大学院情報科学研究科附属組込みシステム研究センター研究員，2008 年同特任助教を経て，2010 年 4 月より同大学院工学研究科講師．組み込みシステム設計，エネルギー最適化等の研究に従事．博士（工学）．IEEE 会員．

高田　広章（正会員）

名古屋大学大学院情報科学研究科情報システム学専攻教授．1988 年東京大学大学院理学系研究科情報科学専攻修士課程修了．同専攻助手，豊橋技術科学大学情報工学系助教授等を経て，2003 年より現職．2006 年より大学院情報科学研究科附属組込みシステム研究センター長を兼務．リアルタイム OS，リアルタイムスケジューリング理論，組み込みシステム開発技術等の研究に従事．オープンソースの ITRON 仕様 OS 等を開発する TOPPERS プロジェクトを主宰．博士（理学）．IEEE，ACM，電子情報通信学会，日本ソフトウェア科学会各会員．