情報処理学会論文誌　Vol. 52　No. 8　2431–2440 (Aug. 2011)

オフセット付きCANメッセージの正確な最大遅れ時間解析

倉地　亮[†1]　陳　暘[†1]　高田広章[†1]

これまでにオフセット付きCANメッセージの最大遅れ時間の解析手法が提案されている．しかしながら，従来の手法では計算量を抑え速く効率的に計算できる一方，悲観的に解析される場合がある．このため，本論では従来手法の問題点を指摘し，正確な最大遅れ時間を求めるための手法を提案する．提案する手法は，最大遅れ時間となりうるメッセージ送信シーケンスに着目し，限定した範囲に限り網羅的に解析を行うことで，正確な最大遅れ時間を導出する方法である．本提案手法の評価として自動車メーカから提供されるメッセージセットに適用した結果，十分に少ない組合せで正確な最大遅れ時間が得られた．

Exact Schedulability Analysis for CAN Messages with Offsets

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

The existing analysis for CAN messages with offsets can compute fast and efficient by reducing the amount of computation. However, it is pessimistic in some cases. The causes of the pessimism is that the cumulative accounting of maximum interference functions is overestimated much greater than a realistic situation. In this paper, we show the cause of this situation that existing analysis takes into account unrealistic interference in the cumulative accounting. Then, we propose the exact schedulability analysis which involves the partially exhaustive analysis. According to our experiments by using actually-used message sets, we can conclude that our approach can significantly reduce the combination of the critical interference function, and its can achieve an exact schedulability analysis.

1. はじめに

車載ネットワークの中でも高いリアルタイム性が要求される制御系ネットワークにおいてCAN（Controller Area Network）[1]が広く使われており，ハードリアルタイムシステムの研究分野においても，CANメッセージの最大遅れ時間を解析する方法が研究されている．CANメッセージの最大遅れ時間解析とは，CANメッセージに対し静的優先度ベーススケジューリングであるRate Monotonic Analysisを適用し，各メッセージが送信要求されてから送信完了するまでの最大実行時間を計算することで，与えられたデッドラインを満たすかどうかを確認するための手法である．

まず，最初のCANメッセージの最大遅れ時間の解析手法として，Tindellらは基本となるオフセットが付かないCANメッセージの解析手法を提案し[2]，これまでに多くの研究が行われた[3)–6)]．その後，各CANメッセージにオフセットと呼ばれる初回送信時までの待ち時間を持たせ，各ECU（Electronic Control Unit）から送信されるCANメッセージを分散させることで，リアルタイム性を保証しながらCANネットワークの回線使用率を向上させる取り組みがなされており[7)–9)]，これらのオフセット付きCANメッセージの最大遅れ時間の解析手法については，飯山らがマルチフレームタスクのモデルを適用した手法を提案した[10]．また，Duらは安全な解析手法として，飯山らとは異なる解析手法を提案している[11]．しかしながら，これらの従来手法は効率良く解析できる一方，実際よりも悲観的に解析されてしまうことがある．

そこで本論では，飯山らが提案する従来手法に着目し，従来手法が悲観的に解析されてしまう場合について，具体例を用いて指摘する．そのうえで，現実的な計算時間で正確な最大遅れ時間を求める手法として，従来手法より求まる最大遅れ時間に影響を与えるcritical interference function（以降，クリティカルIFと呼ぶ）に対してのみ網羅的に組合せ解析を行う方法を提案する．クリティカルIFとは，ある時刻に最も邪魔するCANメッセージのシーケンスを指し，マルチフレームタスクの解析におけるクリティカルフレーム[12]を拡張したものである．これは，CANメッセージの解析の場合にはマルチフレームタスクの解析とは性質と対象が異なるため，文献12)の手法をそのまま適用しても有効な手法とはならないことに起因する．

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

また，オフセットが付く CAN メッセージの解析においては critical instant があらかじめ 1 つに定まらないため，いくつかの critical instant candidate において最大遅れ時間を解析し，その候補の中から最も大きい最大遅れ時間を持つ状況を critical instant として決定する方法をとる[10]．なお，critical instant とは最大遅れ時間が発生する状況を指す．このため，critical instant の送信状況を示すことは，解析結果の妥当性を確認するために重要であるが，従来手法では最大遅れ時間を解析できるものの，その送信状況までは導出することができない．そこで，我々が提案する手法では，網羅的な解析を行うことで critical instant の送信状況を導出する．仮に critical instant の送信状況が分かれば，シミュレータ等を用い，その状況の再現が容易となるため，解析結果の妥当性が確認しやすくなる．

本論の構成は以下のとおりである．2 章で従来の解析手法とその問題を示し，3 章では問題解決のためのアプローチと正確な最大遅れ時間を求める解析手法を提案する．4 章では自動車メーカから提供されたメッセージセットを用いて評価を行い，本手法の有効性を示し，5 章で本論をまとめる．

2. 従来手法

本章では，まず対象となる CAN とそのメッセージモデルの説明を行う．そのうえで，従来手法であるオフセットつき CAN メッセージの最大遅れ時間解析手法について説明し，従来手法が悲観的に解析される状況を指摘する．

2.1 CAN

CAN とは最大伝送速度 1 Mbps とするプロトコルであり，車載ネットワークでは 1 つのバスに複数の ECU が接続されるバストポロジが広く採用されている．CAN では最大 8 バイトのデータを付与することができるフレームと呼ばれるメッセージ単位で通信を行う．各メッセージには 11 ビットのユニークな ID が割り付けられており，バス上での送信権の調停時にはこの ID が優先度を表す．各 ECU はメッセージの送信要求が行われた後，バス上にメッセージが送信されていない場合にはすぐに送信を開始することができる．一方，すでに他の ECU が送信を開始している場合には，その送信されているフレームに同期し自らも送信を開始することで，バス上で ID を用いた送信権の調停が行われる．その送信権の調停の結果，最も優先度の高いメッセージを送信する ECU のみが送信動作を継続し，それ以外の調停に負けた ECU は送信動作を中断し受信動作に移行することで，バス上には最も優先度の高い 1 つのフレームのみが送信される．

2.2 メッセージモデル

一般的な車載ネットワークで用いられるメッセージセットは，複数の ECU [⋆1]で構成されており，各 ECU には複数の送信メッセージが定義されている．各 ECU の送信要求は，初回送信時に限ってはオフセットと呼ばれる初回送信時刻だけ待ってから送信要求が行われ，以降，各 ECU が保持するタイマを用いて周期的に実行される．CAN ではこの送信要求に用いられるタイマを同期させる機構はないため，各 ECU は互いに非同期に送信要求を行う．なお，オフセットの効果は 2 つ存在し，1 つは同じ ECU 内のメッセージには，ECU 内のメッセージの送信要求を分散させることで，各メッセージの最大遅れ時間を低減できる点である．もう 1 つは，他の ECU から送信される低優先度メッセージにとっては，連続して送信される高優先度メッセージが少なくなるため，その低優先度メッセージの最大遅れ時間も低減させることができる点である．

以降では，あるメッセージ τ_i をメッセージの優先度 P_i，最大送信時間 E_i，送信周期 C_i，初回送信時のオフセット O_i の 4 つ組 (P_i, E_i, C_i, O_i) で表すものとする．なお，メッセージの最大送信時間 E_i，送信周期 C_i，オフセット O_i および解析手法により導出される最大遅れ時間は，1 ビットの送信時間の整数倍で表現できるものとする．これは，最大送信時間 E_i は 1 ビットの送信時間を整数倍したものであり，導出される最大遅れ時間も同様となる．また，送信周期 C_i およびオフセット O_i はミリ秒単位で設定されるため，CAN の最大伝送速度である 1 Mbps とした場合の 1 ビットの送信時間である 1 μs に比べ十分に小さいため，1 ビット時間の整数倍と近似できるためである．このメッセージモデルの具体例として，あるメッセージ $\tau_i = (1, 3, 12, 4)$ は**図 1** のように表記できる．

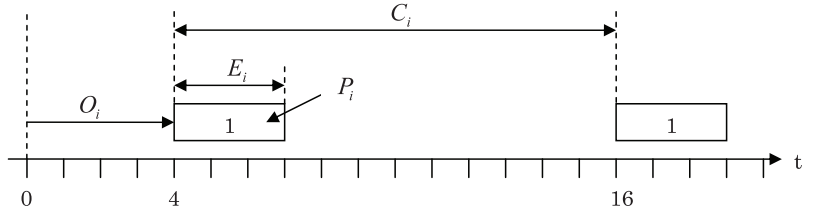


図 1　CAN メッセージの表記
Fig. 1　Notation of CAN messages.

⋆1 K 個の ECU が存在する場合には，ECU_1,...,ECU_K が存在するものとする．

2.3 従来手法の概要

飯山らは，Tindell らの手法が解析対象となるメッセージの送信要求とすべてのメッセージの送信要求が同時に発生する状況を前提としており，オフセットの付いた場合には正確に解析できないことを指摘したうえで，オフセットの付いた CAN メッセージの最大遅れ時間を求める手法を提案した[10]．飯山らが提案する手法では，オフセット付き CAN メッセージのモデルがマルチフレームタスクのモデルと類似しているということから，マルチフレームタスクの解析手法として提案されている Maximum Interference Function（以降，MIF と呼ぶ）を適用した．

2.4 Maximum Interference Function

MIF とは，あるタスクがある時刻において他のタスクを邪魔しうる最大邪魔時間を表す時間関数であり[13]，CAN メッセージの解析においては，ある ECU に存在する対象メッセージの送信を他の ECU から送信される高優先度メッセージ群が最大で邪魔する時間を導出するために用いられる．ここで，MIF の具体例として，2 つのメッセージ $\tau_1 = (1, 1, 6, 0)$ と $\tau_2 = (2, 2, 12, 3)$ が含まれる ECU_J の MIF について，**図 2** を用いて説明する．図 2 は，

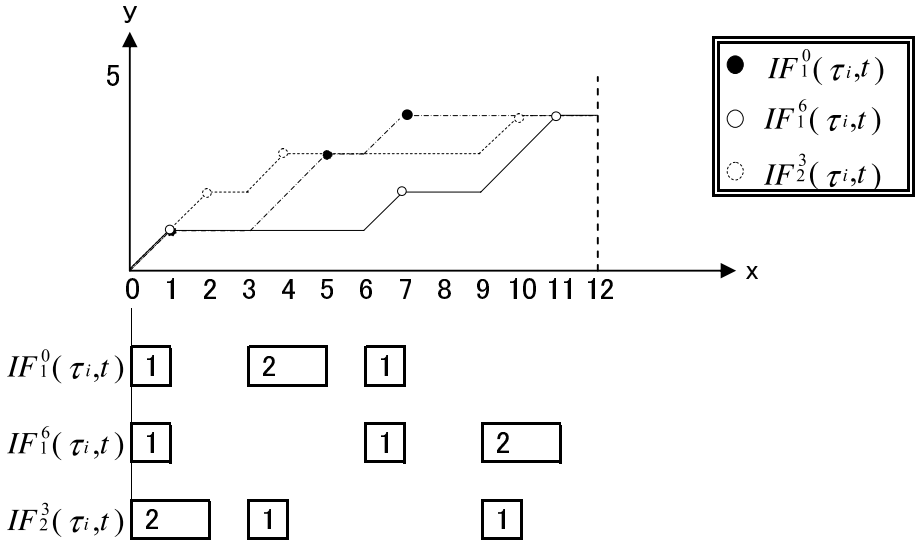


図 2　τ_1 と τ_2 を含む ECU_J の IF と MIF の例
Fig. 2　Examples of MIF and IF of τ_1 and τ_2 in ECU_J.

τ_1 と τ_2 の送信周期の最小公倍数（以降，LCM）である時刻 12 までの間に発生する各メッセージの送信要求時刻からの送信状況を表している．なお，各メッセージの送信要求時刻から始まる状況をクリティカルリリースとし，この各送信要求時刻から始まる Interference function（以降，IF と呼ぶ）を作成し，MIF を導出する．

IF とは，ある開始時刻からの累積的な邪魔時間を表す時間関数であり，CAN メッセージの解析においては対象メッセージよりも優先度が高いメッセージから対象メッセージの送信が邪魔される時間を表すために用いられる．具体的には，対象メッセージ τ_i よりも優先度が高いあるメッセージ τ_j のある送信要求時刻 ST_j を時刻 0 とする時間関数として，あるメッセージ τ_j の優先度 P_j を用いて，$IF_{P_j}^{ST_j}(\tau_i, t)$ と定義できる．また，IF は，x 軸を開始時刻からの相対的な経過時間，y 軸を邪魔時間とする各凸点の座標 (x,y) の順列とその周期 $Cycle$ を用いて，以下の式 (1) のように表記する．

$$IF_{P_j}^{ST_j}(\tau_i, t) = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n), Cycle\} \quad (1)$$

また，IF と同様に，ある ECU_I の MIF も各凸点の座標 (x, y) の順列とその周期 $Cycle$ を用いて，以下の式 (2) のように表記できる．

$$MIF_I(\tau_i, t) = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n), Cycle\} \quad (2)$$

この具体例として，前述した ECU_J を用いて示すと，仮に τ_1 と τ_2 が対象メッセージ τ_i よりも優先度が高いものとする場合には，各 IF は次のとおりになる．τ_1 の送信要求時刻 0 から始まる状況を $IF_1^0(\tau_i, t) = \{(0, 1), (3, 2), (6, 1), 12\}$，$\tau_1$ の送信要求時刻 6 から始まる状況を $IF_1^6(\tau_i, t) = \{(0, 1), (6, 1), (9, 2), 12\}$，$\tau_2$ の送信要求時刻 3 から始まる状況を $IF_2^3(\tau_i, t) = \{(0, 2), (3, 1), (9, 1), 12\}$ の 3 つの IF からなり，これらを傾きが 1 となるグラフで表すと，図 2 に示すとおりになる．

一方，MIF とは各同期されたグループ内に存在するすべての IF を重ね合わせたときの各時間における最大値であり[13]，図 2 の ECU_J の MIF は $MIF_J(\tau_i, t) = max(IF_1^0(\tau_i, t), IF_1^6(\tau_i, t), IF_2^3(\tau_i, t))$ と定義でき，各 IF の y 軸が最大となる線をつないだ $MIF_J(\tau_i, t) = \{(0, 2), (3, 1), (6, 1), 12\}$ と表される．すなわち，MIF とは，ある ECU から送信する対象メッセージ τ_i よりも優先度の高いメッセージ群が他の ECU に存在する対象メッセージ τ_i を最大で邪魔する状況を表す時間関数である．

2.5 飽和加算

従来手法では，すべての他の ECU から送信される高優先度メッセージが対象メッセージに与える影響を累積的に計算するために，各 ECU ごとに導出する MIF を飽和加算し，その邪

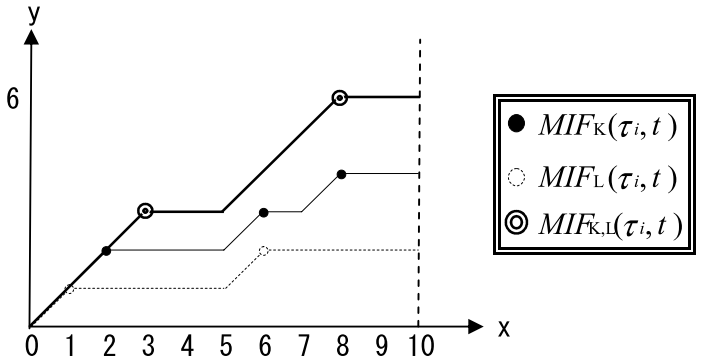


図 3　τ_1 と τ_2 を飽和加算する例
Fig. 3　Examples of a cumulative function.

魔時間の総和を導出している．飽和加算とは，傾きが 1 を超えないように累積的に積算する方法である[14]．**図 3** の例は，ある 2 つの MIF である $MIF_K(\tau_i, t) = \{(0,2),(5,1),(7,1),10\}$ と $MIF_L(\tau_i, t) = \{(0,1),5\}$ を飽和加算し，$MIF_{K,L}$ を生成する状況を表す．2 つの MIF を飽和加算する場合，各 MIF の周期の LCM まで飽和加算することで，飽和加算後の MIF を $MIF_{K,L}(\tau_i, t) = \{(0,3),(5,3),10\}$ と表すことができる．このように加算したい MIF の各凸点が重なる場合には，傾きが 1 になるよう加算することで，そのすべての邪魔時間を足し合わせることが可能となる．

このような演算を用いて，連続する邪魔時間が最初に途切れる時刻を算出することで，解析対象となるメッセージの最大遅れ時刻を求めることができる．ここで，最初にその影響が途切れる時刻を Earliest Idle Time（以下，EIT と呼ぶ）と定義すると，EIT は MIF の傾きが最初に 0 になる時間を表す．すなわち，図 3 の例では，$MIF_{K,L}(\tau_i, t)$ の EIT は $EIT(MIF_{K,L}(\tau_i, t)) = MIF_{K,L}[0].y = 3$ となり，時刻 3 まで邪魔された後に対象メッセージ τ_i が送信できることを表す．

2.6　オフセット付き CAN メッセージの最大遅れ時間解析手法

従来の解析手法では，解析対象となるメッセージ τ_i に着目し，送信する ECU と優先度を基準に，すべてのメッセージを 4 つのグループに分類したうえで，それぞれから受ける邪魔時間を導出している．具体的には，**図 4** で示すように対象メッセージと同じ ECU から送信される高優先度メッセージ群を $ihp(i)$，低優先度メッセージ群を $ilp(i)$，対象メッセージとは異なる ECU から送信される高優先度メッセージ群を $ohp(i)$，低優先度メッセージ

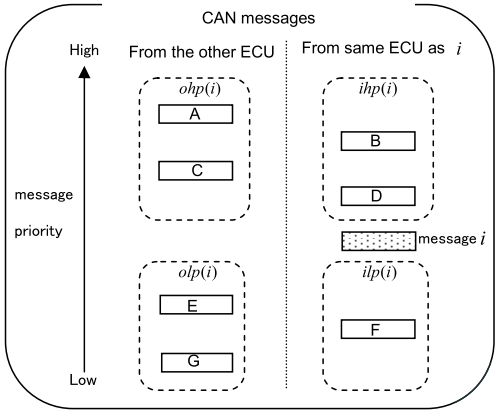


図 4　対象メッセージを基準とするメッセージのグループ分け
Fig. 4　Groups of CAN messages based on priority with message i.

を $olp(i)$ と 4 つに分類する．そのうえで，他の ECU から送信される高優先度メッセージ群 $ohp(i)$ から受ける影響については，各 ECU 単位で MIF を導出したうえですべての MIF を飽和加算しその最大邪魔時間を求める．

ここで，対象メッセージ τ_i の最大遅れ時間を求める方法は以下のとおりになる．なお，ここでは文献 10) 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の解析手法に対し，ECU 内の優先度逆転の考慮を削除し単純化した解析方法[⋆1]ではあるが，この解析方法においても従来手法で発生するのと同様に，悲観的に解析される問題が発生するため，本論では以下の解析方法を従来手法として扱う．

(1)　他 ECU から送信されるすべての高優先度メッセージの最大邪魔時間を求めるために，メッセージ τ_i よりも優先度の高い $ohp(i)$ に対し，各 ECU ごとに MIF を求め，求められた各 ECU の MIF を飽和加算し，すべての他 ECU から送信される高優先度メッセージによる邪魔時間の総和を求める．

(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場合には，低優先度メッセージ 1 メッセージ分に邪魔される．この影響

⋆1 この単純化された解析方法は，送信メールボックスの数が同時に送信要求されるメッセージ数よりも多い場合の解析手法である．

表 1　メッセージセット例
Table 1　Example of a message set.

ECU_I	τ_i	P_i	E_i	C_i	O_i
ECU_1	τ_1	1	1	25	0
	τ_2	2	2	25	5
	τ_3	3	3	25	16
ECU_2	τ_4	4	4	25	0
	τ_5	5	5	25	7
ECU_3	τ_6	6	6	25	0

を考慮するために，他 ECU の低優先度メッセージ $olp(i)$ のうち最長送信時間分の影響を (1) から導出された邪魔時間に飽和加算する．

(3)　最後に自 ECU 内から送信される高優先度メッセージの影響を加算するため，対象メッセージ τ_i のある送信要求時刻から τ_i より高い優先度のメッセージ $ihp(i)$ の IF を作成し，(2) までで求めた邪魔時間と飽和加算し，すべてのメッセージが対象メッセージに与える邪魔時間を求める．このすべてのメッセージからの邪魔時間の最初に傾きが 0 になる時刻まで対象メッセージは邪魔された後，送信を開始することができ，この対象メッセージ τ_i が送信完了した時間が最大遅れ時間の候補の 1 つとなる．

(4)　上記 (3) の計算について，対象メッセージ τ_i とすべての高優先度メッセージの送信要求時刻から計算を行うことで，すべての最大遅れ時間の候補を導出し，その中で最大の遅れ時間を持つ候補が対象メッセージ τ_i の最大遅れ時間となる．

2.7　従来手法の問題点

オフセット付き CAN メッセージの最大遅れ時間を導出する際，従来の手法では MIF を飽和加算するために，現実に起こりうる状況よりも悲観的に解析されている場合が存在する．この具体例について，**表 1** の簡単なメッセージセットを用いて説明する．

表 1 のメッセージセットは，$ECU_1 = \{\tau_1, \tau_2, \tau_3\}$，$ECU_2 = \{\tau_4, \tau_5\}$，$ECU_3 = \{\tau_6\}$ の 3 つの ECU からなる．ここで，ECU_3 から送信されるメッセージ τ_6 を対象メッセージとした場合，他の ECU（つまり，ECU_1 と ECU_2）から送信される高優先度メッセージ群 $ohp(6) = \{\{\tau_1, \tau_2, \tau_3\}, \{\tau_4, \tau_5\}\}$ に対し，各 ECU ごとに MIF を求めたうえですべての MIF を飽和加算することにより，対象メッセージ τ_6 を最大で邪魔する時間を導出する．具体的には，ECU_1 および ECU_2 の MIF はそれぞれ**図 5** と**図 6** に示すとおりであり，$MIF_1(\tau_6, t) = \{(0,3), (9,1), (13,1), (15,1), 25\}$，$MIF_2(\tau_6, t) = \{(0,5), (8,4), 25\}$ となる．このとき，$MIF_1(\tau_6, t)$ と $MIF_2(\tau_6, t)$ を飽和加算すると，$MIF_{1,2}(\tau_6, t) = \{(0,14), (15,1), 25\}$ となり，**図 7** に示すとおり，対象メッセージ τ_6 は最初に傾きが 0 になる時刻 $EIT(MIF_{1,2}(\tau_6, t)) = MIF_{1,2}[0].y = 14$ まで邪魔されてから送信を開始することができる．

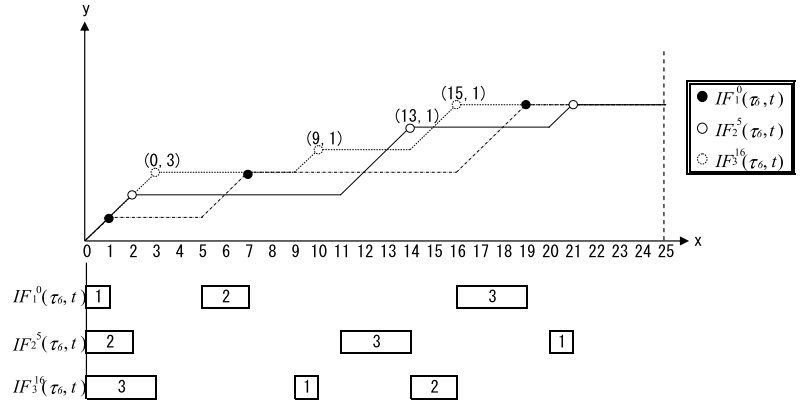


図 5　ECU_1 の MIF
Fig. 5　MIF of ECU_1.

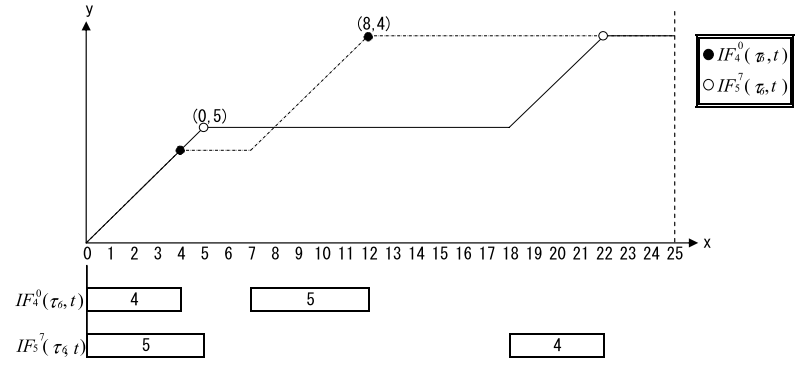


図 6　ECU_2 の MIF
Fig. 6　MIF of ECU_2.

一方，実際に発生しうる最悪状況を調べるためには，文献 12) に定義されるように，すべての ECU に存在する全メッセージの送信要求時刻から始まる組合せにより解析する必要があり，その組合せ数は各 ECU に存在する送信要求時刻の数の直積により導出で

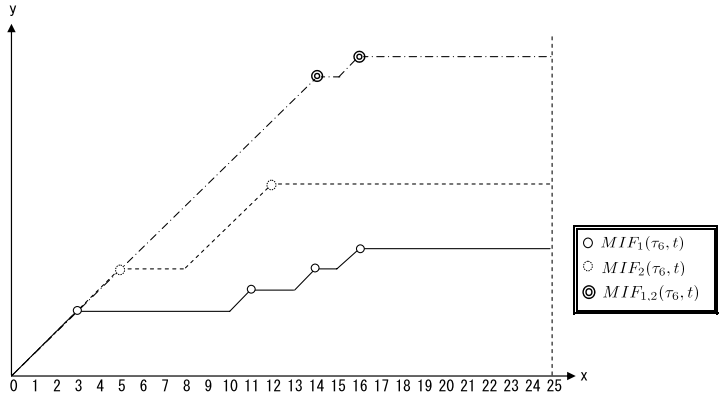


図 7　ECU_1 と ECU_2 の MIF の飽和加算結果
Fig. 7　Cumulative accounting of maximum interference functions ECU_1 and ECU_2.

きる．つまり，表 1 の例においては，ECU_1 に存在する送信要求時刻の数 $\times ECU_2$ に存在する送信要求時刻の数の組合せから，そのすべての状況に対し最大遅れ時間の候補を求め，導出された候補の中の最も大きい遅れ時間を持つ候補が実際に起こりうる最大遅れ時間となる．ここで，表 1 のメッセージセットにおいて対象メッセージ τ_6 を最も邪魔する ECU_1 と ECU_2 の送信状況は，ECU_1 の τ_3 と ECU_2 の τ_4 の送信要求時刻から始まる $IF_3^{16}(\tau_6,t) = \{(0,3),(9,1),(14,2),25\}$ と $IF_4^0(\tau_6,t) = \{(0,4),(7,5),25\}$ を飽和加算した場合であり，これを $max(IF_{ECU_1\times ECU_2}(\tau_6,t))$ と定義すると，**図 8** のとおり $EIT(max(IF_{ECU_1\times ECU_2}(\tau_6,t))) = 13$ だけ対象メッセージ τ_6 を邪魔する場合が最悪となる．これらの結果から，表 1 のメッセージ τ_6 の最大遅れ時間を MIF を飽和加算して解析する場合には，実際に起こりうる最悪状況よりも悲観的に解析されていることが分かる．このように，MIF を飽和加算した結果が実際に起こりうる最悪状況よりも悲観的に解析されるのは，$MIF_1(\tau_6,t)$ の 2 つ目の凸点である $(13,1)$ が $IF_2^5(\tau_6,t) = \{(0,2),(11,3),(20,1),25\}$ のときに発生する状況であり，実際の最悪状況である $IF_3^{16}(\tau_6,t)$ では起こりえないためである．このことより，MIF は各時刻における最大邪魔時間を表しているものの，MIF を飽和加算する場合には，MIF の凸点が実際に起こりうる状況であるかを考慮する必要があるといえる．

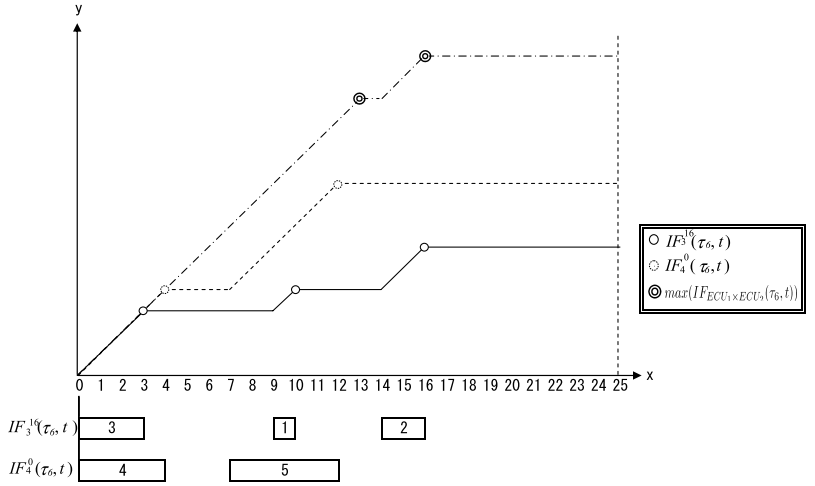


図 8　ECU_1 と ECU_2 の IF の組合せからえられる最悪状況
Fig. 8　Worst case situation of ECU_1 and ECU_2.

3. 提 案 手 法

本章では，正確なオフセット付き CAN メッセージの最大遅れ時間を導出するために，文献 12) で示されている網羅的な組合せによるアルゴリズムを用いて，効率的に解析する方法を提案する．以降では，まず提案手法のアプローチについて説明し，効率的に解析するためのアイデアとなるクリティカル IF と組合せ数の削減方法について言及し，最後に提案する解析手法をまとめる．

3.1 アプローチ

提案手法では，現実的な計算量による組合せ解析を行うために，次の 2 つのアイデアを用いる．1 つは，各 MIF の凸点がどの IF から生成されたかを導出する方法である．仮に各凸点がどの IF からの影響を表すものかが特定できれば，すべての IF に対して網羅的な組合せ解析を行わなくても，特定された IF に対してのみ網羅的に組合せ解析を行うことで計算量を抑えることができる．本論では，この MIF の各凸点を表す IF をクリティカル IF と呼ぶ．もう 1 つは，最大遅れ時間に影響を与えるクリティカル IF のみに限定することで組合せ数を削減する方法をとる．

3.2 クリティカル IF

MIF により邪魔時間を計算する場合，MIF の各凸点を表すクリティカル IF が最大遅れ時間に影響を与える一方で，クリティカル IF ではない IF は最大遅れ時間に影響を与えることはない．クリティカル IF とクリティカル IF ではない IF については，図 5 の例を用いて説明する．図 5 の $MIF_1(\tau_6,t)$ は，MIF の各凸点として $(0,3)$，$(9,1)$，$(13,1)$，$(15,1)$ の 4 点が存在し，$(0,3)$，$(9,1)$，$(15,1)$ の 3 点は $IF_3^{16}(\tau_6,t)$ をクリティカル IF とし，$(13,1)$ の 1 点は $IF_2^5(\tau_6,t)$ をクリティカル IF とする MIF である．一方，MIF のどの凸点にもなりえない $IF_1^0(\tau_6,t)$ はクリティカル IF ではない IF であるため，$IF_1^0(\tau_6,t)$ を用いて組合せ解析をしても最悪とはなりえない．これは，すべての送信要求時刻の組合せ数よりもクリティカル IF のみに限定した場合の方が組合せ数が減る効果を表している．

3.2.1 クリティカル IF の抽出方法

クリティカル IF の抽出方法について，IF および MIF の生成方法を用いて説明する．まず，クリティカル IF を抽出するために，各 IF および MIF に以下のような開始時刻情報を付与する．開始時刻情報とは，IF を生成する際の開始メッセージ τ_j の優先度 P_j と送信要求時刻 ST_j の 2 つ組 (P_j,ST_j) で表すものとし，ある対象メッセージ τ_i に対する IF の各凸点に以下のように付与されるものとする．

$$IF_{P_j}^{ST_j}(\tau_i,t)=\{(x_1,y_1,(P_j,ST_j)),(x_2,y_2,(P_j,ST_j)),...,(x_n,y_n,(P_j,ST_j)),Cycle\} \tag{3}$$

また，各 MIF においては，1 つの凸点に複数の開始時刻情報を含む場合が存在するため，各凸点に含まれる開始時刻情報群を $\langle\rangle$ で囲み，ある対象メッセージ τ_i に対する ECU_I の MIF は以下のように表記する．

$$MIF_I(\tau_i,t)$$
$$=\{(x_1,y_1,\langle(P_j,ST_j)\rangle),(x_2,y_2,\langle(P_j,ST_j)\rangle),...,(x_n,y_n,\langle(P_j,ST_j)\rangle),Cycle\} \tag{4}$$

これらの表記を用いて，以下のように IF や MIF を生成し飽和加算することで，クリティカル IF を抽出する．

(1) 与えられたメッセージ情報から IF を生成する際，IF のすべての凸点には開始時刻情報を付与する．図 5 の例を用いて具体的に説明すると，各 IF は $IF_1^0(\tau_6,t)=\{(0,1,(1,0)),(5,2,(1,0)),(16,3,(1,0)),25\}$，$IF_2^5(\tau_6,t)=\{(0,2,(2,5)),(11,3,(2,5)),(20,1,(2,5)),25\}$，$IF_3^{16}(\tau_6,t)=\{(0,3,(3,16)),(9,1,(3,16)),(14,2,(3,16)),25\}$ と表すものである．

(2) 各 IF から MIF を生成する際，すべての IF を重ね合わせ，MIF の凸点となりうる開始時刻情報のみを残し，それ以外の開始時刻情報はすべて破棄する．この操作により，図 5 の MIF_1 は $MIF_1(\tau_6,t)=\{(0,3,\langle(3,16)\rangle),(9,1,\langle(3,16)\rangle),(13,1,\langle(2,5)\rangle),(15,1,\langle(3,16)\rangle),25\}$ となり，同様に図 6 の MIF_2 は $MIF_2(\tau_6,t)=\{(0,5,\langle(5,7)\rangle),(8,4,\langle(4,0)\rangle),25\}$ となる．

(3) MIF および IF を飽和加算する場合，MIF や IF の各凸点に含まれるすべての開始時刻情報を残したまま，凸点を生成する．すなわち，図 5 の $MIF_1(\tau_6,t)$ と図 6 の $MIF_2(\tau_6,t)$ を飽和加算した場合には，$MIF_{1,2}(\tau_6,t)=\{(0,14,\langle(3,16),(2,5),(5,7),(4,0)\rangle),(15,1,\langle(3,16)\rangle),25\}$ となり，$MIF_{1,2}(\tau_6,t)$ のすべての凸点に含まれる開始時刻情報からすべてのクリティカル IF を抽出することができる．

3.3 組合せ数の削減方法

より効率的に組合せ解析を行うために，MIF に存在するすべてのクリティカル IF ではなく，最大遅れ時間に影響を与えるクリティカル IF のみに限定する方法をとる．具体的には，従来手法を用いて最大遅れ時間を求め，その最大遅れ時間に影響を与えた MIF の凸点に存在する開始時刻情報のみを抽出する方法である．これは，一般的な車載ネットワークに用いられるメッセージセットが，全体で数十個の送信メッセージが存在するうえ，それらのメッセージは優先度に応じて送信周期がゆるやかに大きくなるよう与えられていることが多いために，各 ECU に存在する送信メッセージの送信周期の LCM が大きくなり，MIF に含まれる各凸点の数も非常に多くなるためである．このため，各 ECU に存在する送信メッセージの送信周期の LCM は表 1 の例のようにすべてのメッセージで送信周期が一致することは稀で，実際には各メッセージの送信周期よりも非常に大きい LCM になることが多く，その LCM までに存在するすべてのクリティカル IF に対して網羅的に解析することは計算量の観点から無駄が多い．このため，より効率的な解析とするためには，最大遅れ時間に影響を与えるクリティカル IF のみに限定する方法が有効と考えられる．

3.4 オフセット付き CAN メッセージの正確な最大遅れ時間解析手法

本提案手法では，大きく 2 ステップに分けて解析を行う．まず最初に，前述した最大遅れ時間に影響を与えるクリティカル IF を抽出するために，従来手法を用いて最大遅れ時間を求める．その後，従来手法から導出された最大遅れ時間に影響を与えるクリティカル IF のみを用いて網羅的に組合せ解析を行う方法である．この具体的な解析方法を以下に示す．

(1) ある対象メッセージを解析する際，最大遅れ時間に影響を与えるクリティカル IF の

みを抽出するために，2.6 節で提示する従来手法を用いて最大遅れ時間を求める．このとき，3.2.1 項で定義する 3 つの処理を用いて最大遅れ時間を求めることで，対象メッセージの最大遅れ時間に影響を与える高優先度メッセージの開始時刻情報を抽出する．

(2) 抽出された開始時刻情報に対して，網羅的な組合せ解析を実行する．網羅的な組合せ解析とは，抽出された開始時刻情報を各 ECU に分類したうえで，すべての ECU に存在する開始時刻情報から始まる IF の組合せから最大遅れ時間を導出する方法である．この網羅的な組合せ解析の過程で，(1) から得られた最大遅れ時間と一致する場合には，その IF の組合せ状況を critical instant として出力し，次のメッセージの解析へと移る．一方，抽出されたすべての IF の組合せから最大遅れ時間を導出した結果，(1) から得られた最大遅れ時間と一致しない場合には，(2) で得られた最も大きい最大遅れ時間を持つ IF の組合せ状況を critical instant として出力する．

(3) 解析するすべてのメッセージに対して上記 (1) と (2) を実行し，解析するメッセージがなくなったら終了する．

4. 評　　価

本評価では，自動車メーカから提供されたメッセージセットを用いて，提案手法と従来手法とを比較した．本章では，まず解析の対象と前提について説明する．次に評価結果として，解析に要する計算時間と解析結果を従来手法と比較し，そのうえで，提案手法の有効性として組合せ数の削減量について示す．

4.1 対象と前提

評価で使用したメッセージセットの概要は，CAN の転送速度 500 kbps，ECU 数 14 個，メッセージ数 64 メッセージ，バス負荷率は約 60%である．バス負荷率とはすべての ECU から送信されるメッセージの負荷率の合計であり，送信メッセージの負荷率とは，ある送信メッセージ τ_i の最大送信時間 E_i をその送信周期 C_i で割り百分率で表した値のことを指す．

また，評価環境として，CPU：Intel® Core2 Quad 2.83 GHz，OS：Microsoft Windows XP sp3，メモリ：3.25 GB の PC を用いた．

4.2 評価結果 1：従来手法と提案手法の計算時間の比較

我々が提案する手法では，解析結果が厳密となり，最大遅れ時間が発生する状況が導出できるものの，従来手法に比べ計算量が増加することが懸念される．このため，解析に要する計算時間を従来手法と比較し，提案手法が現実的な時間内に解析できることを検証する．

表 2　従来手法との計算時間の比較
Table 2　Comparison of computation times of existing analysis and proposed analysis.

	従来手法	提案手法
計算時間	1.719 秒	727.520 秒

なお，現実的な時間内とは，本解析ソフトウェアを実際に自動車メーカで運用する際に，使用に耐えうる程度の計算時間で解析が終了することを指す．

従来手法と提案手法の解析に要する計算時間の比較結果は**表 2** に示すとおりであり，提案手法は，従来手法に比べて計算時間が長くなるものの，現実的な時間内にすべてのメッセージの解析を完了することを示している．

4.3 評価結果 2：従来手法と提案手法の解析結果の差

従来手法で解析する場合には 64 メッセージ中 10 メッセージが悲観的に解析されていた．この悲観的に解析されたメッセージ中，最大で約 7%も遅れ時間が過大に評価されているメッセージが存在した．

4.4 評価結果 3：組合せ数の削減量

さらに，我々の手法の有効性を示すために，他の ECU から送信される高優先度メッセージを網羅的に解析する際の組合せ数について，全送信要求時刻の組合せ数（全 IF の組合せ数）と全クリティカル IF の組合せ数と比較した．評価結果の具体例として，提案手法において最大の組合せ数を持つ CAN メッセージの各 ECU における組合せ数と各 ECU の組合せ数を直積した総組合せ数の比較結果を**表 3** に示す．表 3 の結果より，提案手法により限定されたクリティカル IF の総組合せ数は全送信要求時刻の総組合せ数よりも十分に少なく，全クリティカル IF の数に対し 7.36×10^{-11} 倍以下に組合せ数を減らすことができており，十分に計算量を削減できているといえる．

同様に，提供されたメッセージセット内の全 64 メッセージに対し，総組合せ数の比較を行った．この評価結果として，提案手法による総組合せ数の削減量ごとに全 64 メッセージを分類した結果を**表 4** に示す．この結果から，組合せ数が減らないメッセージが数個存在するものの，これらは優先度の高いメッセージであるために組合せ数を減らさなくても十分に少ない組合せ数で解析できるため，これ以上組合せ数を減らすことができないメッセージといえる．一方，それ以外のすべてのメッセージについては組合せ数を減らすことができ，特に優先度が低いメッセージについては組合せ数を 1/10000 以下に減らすことができたメッセージ数が，全送信要求時刻との比較では 30 メッセージ，全クリティカル IF との比較では 10 メッセージ存在し，メッセージセット全体に対しても十分な計算量の削減がで

表 3　提案手法において最大の組合せ数を持つ CAN メッセージの組合せ数の比較
Table 3　Comparison of the number of combinations in the evaluation.

ECU_i	全送信要求時刻	全クリティカル IF	提案手法
ECU_1	616	89	8
ECU_2	421	150	5
ECU_3	141	50	4
ECU_4	3	3	2
ECU_5	26	16	2
ECU_6	1	1	1
ECU_7	11	10	1
ECU_9	621	142	6
ECU_{10}	111	12	2
ECU_{11}	101	51	2
ECU_{12}	6	5	1
ECU_{13}	1	1	1
ECU_{14}	31	3	2
総組合せ数	4.06×10^{19}	4.177×10^{14}	30,720

表 4　各メッセージの組合せ数の削減量による分類
Table 4　Classification of the reduction of combination number for each message.

提案手法による 組合せ数の削減量	全送信要求時刻との比較	全クリティカル IF との比較
1（削減なし）	2 メッセージ	4 メッセージ
1 より少ない〜1/100 未満	24 メッセージ	31 メッセージ
1/100〜1/10000 未満	8 メッセージ	19 メッセージ
1/10000 以下	30 メッセージ	10 メッセージ

きていることを示している．この結果より，提案手法の組合せ数が全送信要求時刻や全クリティカル IF を用いた組合せ数に比べ，十分に計算量を減少させており，本手法が有効であるといえる．

5. ま と め

本論では，従来の手法では悲観的に解析される具体例を示したうえで，効率良く正確な遅れ時間を解析するための手法を提案した．具体的には，限られた範囲のクリティカル IF に着目し，計算量を減らしたうえで網羅的な組合せ解析を行う手法である．この手法の評価として，自動車メーカから提供されるメッセージセットに適用した結果，従来の手法では悲観的に見積もられるメッセージが 64 メッセージ中 10 メッセージ存在しており，従来手法よりは計算時間が必要ではあるものの，本論で提案する手法を用いることで，現実的な時間内に正確な最大遅れ時間を得ることができた．また，これまでに従来手法がどれだけ悲観的に解析されているかを示す手法はなかったが，本提案手法を用いることで従来手法がどれだけ悲観的に解析されているかが明らかになった．さらに，我々の手法では網羅的な解析を行うことで，従来手法では示すことができない各メッセージの最大遅れ時間の発生状況を導出することができた．

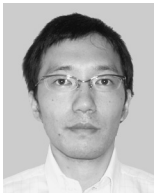
謝辞　本研究を進めるにあたり，貴重なご意見をいただいたスズキ株式会社の堂畑克彦氏，増田達彦氏に厚く御礼申し上げます．

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) Tindell, K. and Burns, A.: Guaranteed Message Latencies for Distributed Safety-Critical Hard Real-Time Networks, Technical Report YCS 229, Department of Computer Science, University of York (1994).
3) Tindell, K., Burns, A. and Wellings, A.: Calculating Controller Area Network (CAN) message response times, *Control Engineering Practice*, Vol.3, No.8, pp.1163–1169 (1995).
4) Punnekkat, S., Hansson, H. and Norström, C.: Response time analysis under errors for CAN, *Proc. 6th IEEE Real-Time Technology and Applications Symposium* (*RTAS*), Washington DC, IEEE Computer Society Press (2000).
5) Nolte, T.: Reducing pessimism in CAN response time analysis, Mälardalen University, Mälardalen Real-Time Research Centre, Technical Report MDH-MRTC-51 (2002).
6) 飯山真一，高田広章：システム構成を考慮した CAN の最大遅れ時間解析手法，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.SIG1(ACS4), pp.66–76 (2004).
7) Grenier, M., Goossens, J. and Navet, N.: Near-optimal fixed priority preemptive scheduling of offset free systems, *Proc. 14th International Conference on Network and Systems* (*RTNS 2006*), Poitiers, France, May 30–31 (2006).
8) 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*) (2008).
9) 陳　暘，倉地　亮，曾　剛，高田広章：CAN メッセージのオフセット決定手法，情報処理学会論文誌，Vol.52, No.7, pp.2245–2255 (2011).
10) 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

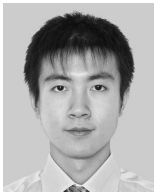
ジの最大遅れ時間解析，情報処理学会論文誌：コンピューティングシステム，Vol.45, No.SIG11(ACS 7), pp.455–464 (2004).
11) Du, L. and Xu, G.: Worst Case Response time analysis for CAN messages with offsets, *Proc. International Conference on Vehicular Electronics and Safety* (*ICVES 2009*), pp.41–45 (2009).
12) Zuhily, A. and Burns, A.: Exact schedulability analysis of non-accumulatively monotonic multiframe tasks, *Real-Time Syst.*, Vol.43, No.2, pp.119–146 (2009).
13) 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).
14) 飯山真一，高田広章，菅沼英明：エンジン制御システムのためのリアルタイム性検証手法，情報処理学会論文誌，Vol.43, No.6, pp.1715–1724 (2002).

(平成 22 年 12 月 24 日受付)
(平成 23 年 5 月 14 日採録)



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，同年より現職．車載ネットワークに関する研究に従事．



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）．

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．