

(19) 日本国特許庁(JP)

(12) 公表特許公報(A)

(11) 特許出願公表番号

特表2012-518215

(P2012-518215A)

(43) 公表日 平成24年8月9日(2012.8.9)

(51) Int.Cl. F I テーマコード (参考)
G06F 11/34 (2006.01) G06F 11/34 S 5B042

審査請求 未請求 予備審査請求 有 (全 26 頁)

(21) 出願番号	特願2011-549577 (P2011-549577)	(71) 出願人	511125308
(86) (22) 出願日	平成22年2月12日 (2010.2.12)		インクロン ゲーエムベーハー
(85) 翻訳文提出日	平成23年10月13日 (2011.10.13)		INCHRON GMBH
(86) 国際出願番号	PCT/EP2010/051783		ドイツ ポツダム アウグストーベール
(87) 国際公開番号	W02010/092146		ーストラッセ 88
(87) 国際公開日	平成22年8月19日 (2010.8.19)		August-Bebel-Strass
(31) 優先権主張番号	61/152,828		e 88 Potsdam German
(32) 優先日	平成21年2月16日 (2009.2.16)		y
(33) 優先権主張国	米国 (US)	(74) 代理人	110000154
			特許業務法人はるか国際特許事務所
		(72) 発明者	アルバース カルステン
			ドイツ ヴァイルハイム イベルハストラ
			ッセ 6

最終頁に続く

(54) 【発明の名称】 システムの実時間処理能力の解析方法

(57) 【要約】

本発明は、システム、特にさまざまなタスクが設けられたコンピュータシステムの実時間を解析する方法であって、タスクは繰り返し実行され、タスクの実行は、タスクの起動によりトリガーされ、これはタスクのイベントを示し、複数の記述的エレメントが前記イベントの時間相関をイベントストリームとして記述するために設けられ、イベントストリームはイベントの最大時間密度および/又は最小時間密度を検出することができ、少なくとも、イベントストリームの量が割り当てられ、少なくとも2つのイベントストリームによりキャプチャされるイベント全体の時間相関を記述するさらなる記述的エレメントを有する方法を提供する。

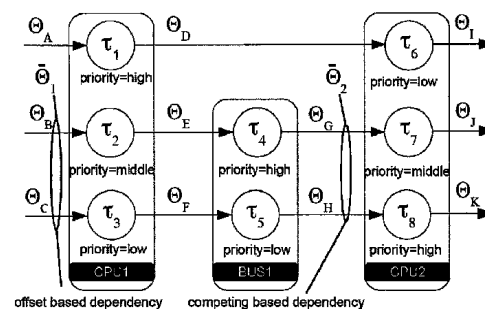


Fig. 1

【特許請求の範囲】**【請求項 1】**

システム、特にさまざまなタスクが設けられたコンピュータシステムの実時間を解析するための、コンピュータに組み込まれる方法であって、

前記タスクは繰り返し実行され、

タスクの実行は、前記タスクの起動によりトリガーされ、これは前記タスクのイベントを示し、

複数の記述的エレメントが前記イベントの時間相関をイベントストリームとして記述するために設けられ、

前記イベントストリームは前記イベントの最大時間密度および / 又は最小時間密度を検出することができ、

少なくとも、イベントストリームの量が割り当てられ、少なくとも 2 つのイベントストリームによりキャプチャされるイベント全体の時間相関を記述するさらなる記述的エレメントを有することを特徴とする方法。

10

【請求項 2】

請求項 1 に記載の方法であって、

前記少なくとも 1 つのさらなる記述的エレメントが限定イベントストリームを記述することを特徴とする方法。

【請求項 3】

請求項 1 又は 2 に記載の方法であって、

前記さらなる記述的エレメントもしくは限定イベントストリームの少なくとも 1 つが、少なくとも 2 つのイベントストリームによりキャプチャされるイベント全体の最大時間相関を記述し、

20

少なくとも時間インターバル ($d t$) において前記限定イベントストリームにより許容されるイベントの量である時間インターバル ($d t$) のイベント量が、 $d t$ において非限定イベントストリームにより許容されるイベントの総量より少ないことを特徴とする方法。

【請求項 4】

請求項 1、2、3 のいずれかに記載の方法であって、

前記さらなる記述的なエレメント又は限定イベントストリームの少なくとも 1 つが、少なくとも 2 つのイベントストリームによりキャプチャされるイベント全体の最小時間相関を記述し、

30

少なくとも時間インターバル ($d t$) に対し、 $d t$ において前記限定イベントストリームにより許容されるイベントの量が、 $d t$ において前記非限定イベントストリームにより許容されるイベントの量を上回ることを特徴とする方法。

【請求項 5】

先に記載された請求項のいずれかに記載の方法であって、

前記システムには少なくとも 1 つのコンポーネント又はリソースが割り当てられ、前記コンポーネントの少なくとも 1 つには少なくとも 2 つのタスクが割り当てられることを特徴とする方法。

40

【請求項 6】

先に記載された請求項のいずれかに記載の方法であって、

少なくとも 2 つのタスクが 1 つのグループに含まれ、グループの前記タスクはリソースに割り当てられ、前記リソースにおいて競合する前記タスクもしくはこのグループのタスクは、異なる時間においてのみ前記リソースにおける処理を行い、前記システムは前記リソースが前記グループの所定のタスクに割り当てられる時間点を決定する方法を有することを特徴とする方法。

【請求項 7】

先に記載された請求項のいずれかに記載の方法であって、

前記限定イベントストリームの少なくとも 1 つが少なくとも 2 つの限定イベントストリ

50

ームに割り当てられ、それらはそれぞれ異なるタスクを起動し、
これらのタスクは少なくとも1つの結合グループに割り当てられ、
コストがそれぞれ、1つのイベントのための前記タスクによる前記リソースの一時的な
使用をキャプチャする前記タスクに割り当てられ、
さらに、

前記タスクにより結合して要求される前記コストを決定するステップ、を有し、
該ステップは、

限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当
てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動す
る前記イベントストリームにより限定されるステップ、

前記割り当てられたイベントの量を考慮して前記限定イベントストリームの残りの結果
を決定するステップ、

前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タス
クに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タス
クを起動する前記イベントストリームにより限定されるステップ、

前記ステップを前記限定イベントストリームの全てのイベントが割り当てられるまで繰
り返すステップ、

個々のタスクのコストおよびそれぞれ割り当てられるイベントの量から全体コストを検
出するステップ、を有することを特徴とする方法。

【請求項 8】

システム、特に複数の異なるタスクが設けられたコンピュータシステムの実時間を解析
するための、コンピュータに組み込まれる方法であって

前記タスクは繰り返し実行され、

タスクの実行は、前記タスクの起動によりトリガーされ、これは前記タスクのイベント
を示し、

1又は複数の記述的エレメントが前記イベントの時間相関をイベントストリームとして
記述するために設けられ、

前記イベントストリームは前記イベントの最大時間密度および/又は最小時間密度を検
出することができ、

少なくとも、イベントストリームの量が割り当てられ、少なくとも2つのイベントスト
リームによりキャプチャされるイベント全体の時間相関を記述するさらなる記述的エレ
メントを有し、

前記限定イベントストリームの少なくとも1つが少なくとも2つの限定イベントストリ
ームに割り当てられ、それらはそれぞれ異なるタスクを起動し、

これらのタスクは少なくとも1つの結合グループに割り当てられ、

タスクには、それぞれ1つのイベントのための前記タスクによる前記リソースの一時的
な使用をキャプチャする各コストが割り当てられ、

さらに、

前記タスクにより結合して要求される前記コストを決定するステップ、を有し、
該ステップは、

限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当
てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動す
る前記イベントストリームにより限定されるステップ、

前記割り当てられたイベントの量を考慮して前記限定イベントストリームの残りの結果
を決定するステップ、

前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タス
クに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タス
クを起動する前記イベントストリームにより限定されるステップ、

前記ステップを前記限定イベントストリームの全てのイベントが割り当てられるまで繰
り返すステップ、

10

20

30

40

50

個々のタスクのコストおよびそれぞれ割り当てられるイベントの量から全体コストを検出するステップ、を有することを特徴とする方法。

【請求項 9】

請求項 7 又は 8 に記載の方法であって、

少なくとも 2 つのイベントストリームに、2 つの異なる限定イベントストリームが割り当てられ、さらに、

前記タスクにより結合して要求される前記コストを決定するステップを有し、

該ステップは、

限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動する前記イベントストリームにより限定され、イベントの最小量はそれぞれ前記限定イベントストリームにより許容されるものであるステップ、

前記限定イベントストリームの残りのイベントを決定するステップであって、実際に割り当てられたイベントの量が考慮されるステップ、

前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タスクに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タスクを起動する前記イベントストリームにより限定され、他の限定イベントストリームが前記タスクに割り当てられるステップ、

前記ステップを前記限定イベントストリームの全てのイベントが割り当てられる、もしくはイベント量が前記グループの全てのタスクに割り当てられるまで繰り返すステップ、

個々のタスクのコストおよびそれらにそれぞれ割り当てられるイベントの量から全体コストを決定するステップ、を有することを特徴とする方法。

【請求項 10】

先に記載された請求項のいずれかに記載の方法であって、

前記限定イベントストリームが、時間インターバルにおけるイベントの密度を記述するポイントの量に割り当てられ、前記限定イベントストリームは、イベントに対し異なるポイントの量を要求する少なくとも 2 つの限定イベントストリームに割り当てられる

ことを特徴とする方法。

【請求項 11】

先に記載された請求項のいずれかに記載の方法であって、

第 1 のタスクの前記起動が第 2 のタスクにより実行されることができ、

これらの起動の時間相関は、イベントストリーム（出力される前記第 1 のタスクのイベントストリーム、入力される前記第 2 のタスクのイベントストリーム）によってもキャプチャ可能であることを特徴とする方法。

【請求項 12】

先に記載された請求項のいずれかに記載の方法であって、

少なくとも 2 つのリソースが前記システムに割り当てられ、

タスクの第 1 のグループが第 1 のリソースに割り当てられ、タスクの第 2 のグループが第 2 のリソースに割り当てられ、

前記第 1 のグループの少なくとも 2 つのタスクがそれぞれ前記第 2 のグループのタスクを直接、もしくは複数のタスクを経由して起動する方法であって、

出力される前記第 1 のタスクのイベントストリームに対し、少なくとも限定イベントストリームを決定するステップ、

入力される前記第 2 のタスクのイベントストリームに対し、少なくとも第 2 のタスクに対する起動が第 1 のタスクにより直接実行されない限り、少なくとも限定イベントストリームを決定するステップ、

前記第 2 のタスクの、前記入力されるイベントストリームの前記限定イベントストリームを考慮して、前記第 2 のタスクにより結合的に要求されるコストを検出するステップ、を有することを特徴とする方法。

【請求項 13】

請求項 1 2 に記載の方法であって、少なくとも 1 つの前記第 2 のタスクの前記限定イベントストリームを決定するために、前記第 1 のタスクの前記限定イベントストリームが使用されることを特徴とする方法。

【請求項 1 4】

先に記載された請求項のいずれかに記載の方法であって、

前記タスクは、CPU 又は電気回路モジュールにより実行可能なユニットであることを特徴とする方法。

【発明の詳細な説明】

【技術分野】

【0001】

10

本発明は、コンピュータに組み込まれる方法であって、プログラムモジュール（タスク）および／又は電気回路モジュールを有する複分散システムの実時間応答の解析方法に関する。

【背景技術】

【0002】

このようなシステムは、プログラムモジュール（タスク）および／又は電気回路モジュールの形の、複数のコンポーネントからなる。コンポーネントは互いに通信を行いデータの処理もしくは変換を行う。さらに、そのようなシステムは、データを検出するセンサー、ユーザとの対話を行うモジュール、環境や他のシステムへ物理的な影響を及ぼすアクチュエータ、ユーザや環境に対し情報を伝達する出力メディアを有してもよい。

20

【0003】

特定のコンポーネント（実行リソース）は、部分的に、1 以上のプログラムモジュールを順に実行してもよい。より多くのモジュールが実行される場合、順序やそれぞれの実行時間は、シーケンスプランニングもしくはスケジューリングの手法により各モジュールに割り当てられる。

【発明の概要】

【発明が解決しようとする課題】

【0004】

現在実行できるスケジューリング手法として静的スケジューリング法や動的スケジューリング法がある。静的スケジューリング法では、実行されるモジュールに対する実行順序や実行時間の割り当ては静的に決定される。動的スケジューリング法では、実行されるモジュールに対する実行順序や実行時間の割り当ては、動作時間を考慮する所定の方法により決定される。動作時間は処理されるデータ、他のモジュールによる実行、環境、および／又はユーザとの相互作用に依存する。

30

【0005】

このようなシステムやコンポーネントは、それらが正しい結果や制御決定が求められるだけでなく、検出期間や、結果を出力するタイミングに関する要求も課せられるために、時間制約が厳しいことがある。そのような要求の例として、結果の決定のための最大待ち期間や、数値又はセンサー又は入力データの変化への保証応答時間がある。しかしながら、このような要求は規則的なタイムラグや一定のタイムフレームにおける新たな演算を求

40

【0006】

モジュールはお互いに、通信リンク経由で、起動及び／又は非同期的なデータのやり取りを行うことがある。この通信もしくは起動のリンクのモデリングのために、多数のイベントモデルが開発されている。このような関係は、例えば単一周期、1 周期およびジッタ、イベントストリーム [Gresser: Echtzeitnachweis Ereignisgesteuerter Realzeitsysteme, Munchen 1993, (ここで参照することで包含されとする)]、又は階層イベントストリーム (WO 2008/003427 参照) によりモデル化されることがある。他にも多くのイベントモデルが利用可能である。

【0007】

50

実時間システムの解析に関しては、可能な限り実際のシステムにおける処理に近づけることが重要である。実際の組み込みシステムにおいては、多くのコンポーネントやタスクは暗黙のうちに、実行シナリオとは別に非常に強い相互関係がある。この相互関係は互いの依存関係を考慮することなくして無視することはできない。このように、実時間解析は、実際のシステムよりも長い最大反応時間（応答時間）といった、より保守的な結果を生成しうる。

【 0 0 0 8 】

一連の依存関係は、既存の実時間解析において考慮される。しかしながら、所望のタイプの依存関係を正確に考慮しようとする常により、解析アルゴリズムの、特定の拡張が必要になる。その結果、解析において新たなタイプの依存関係を考慮することは非常に複雑な処理になる。さらに、既存の実時間解析方法は、依存関係の一部しか対応していない。

10

【 0 0 0 9 】

よって、実際の実時間解析から依存関係の決定やモデリングを切り離すメカニズムを持つことは非常に有利なことである。その結果、実時間解析においては、抽象的な依存関係モデルが決定される全ての依存関係を考慮するためには、この抽象的な依存関係モデルを考慮しなくてはならない。

【 0 0 1 0 】

本発明によれば、このようなメカニズムである、イベントストリームを制限する方法を提案する。

【課題を解決するための手段】

20

【 0 0 1 1 】

第 1 の側面によれば、本発明は、システム、特にさまざまなタスクが設けられたコンピュータシステムの実時間を解析するための、コンピュータに組み込まれる方法であって、前記タスクは繰り返し実行され、タスクの実行は、前記タスクの起動によりトリガーされ、これは前記タスクのイベントを示し、複数の記述的エレメントが前記イベントの時間相関をイベントストリームとして記述するために設けられ、前記イベントストリームは前記イベントの最大時間密度および / または最小時間密度を検出することができ、少なくとも、イベントストリームの量が割り当てられ、少なくとも 2 つのイベントストリームによりキャプチャされるイベント全体の時間相関を記述するさらなる記述的エレメントを有することを特徴とする方法が提供される。

30

【 0 0 1 2 】

前記少なくとも 1 つのさらなる記述的エレメントは、限定イベントストリームを記述する。

【 0 0 1 3 】

前記さらなる記述的エレメントもしくは限定イベントストリームの少なくとも 1 つが、少なくとも 2 つのイベントストリームによりキャプチャされるイベント全体の最大時間相関を記述し、少なくとも時間インターバル (dt) において前記限定イベントストリームにより許容されるイベントの量である時間インターバル (dt) のイベント量が、 dt において非限定イベントストリームにより許容されるイベントの総量より少ないことが好ましい。

40

【 0 0 1 4 】

前記さらなる記述的エレメント又は限定イベントストリームの少なくとも 1 つが、少なくとも 2 つのイベントストリームによりキャプチャされるイベント全体の最小時間相関を記述し、少なくとも時間インターバル (dt) に対し、 dt において前記限定イベントストリームにより許容されるイベントの量が、 dt において前記非限定イベントストリームにより許容されるイベントの量を上回る。

【 0 0 1 5 】

前記システムには少なくとも 1 つのコンポーネント又はリソースが割り当てられ、前記コンポーネントの少なくとも 1 つには少なくとも 2 つのタスクが割り当てられることが好ましい。

50

【 0 0 1 6 】

少なくとも2つのタスクが1つのグループに含まれることが好ましく、グループの前記タスクはリソースに割り当てられ、前記リソースにおいて競合する前記タスクもしくはこのグループのタスクは、異なる時間においてのみ前記リソースにおける処理を行い、前記システムは前記リソースが前記グループの所定のタスクに割り当てられる時間点を決定する方法を有する。

【 0 0 1 7 】

本発明の好適な実施形態によれば、前記限定イベントストリームの少なくとも1つが少なくとも2つの限定イベントストリームに割り当てられ、それらはそれぞれ異なるタスクを起動し、これらのタスクは少なくとも1つの結合グループに割り当てられ、コストがそれぞれ、1つのイベントのための前記タスクによる前記リソースの一時的な使用をキャプチャする前記タスクに割り当てられ、

さらに、

前記タスクにより結合して要求される前記コストを決定するステップ、を有し、

該ステップは、

限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動する前記イベントストリームにより限定されるステップ、

前記割り当てられたイベントの量を考慮して前記限定イベントストリームの残りの結果を決定するステップ、

前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タスクに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タスクを起動する前記イベントストリームにより限定されるステップ、

前記ステップを前記限定イベントストリームの全てのイベントが割り当てられるまで繰り返すステップ、

個々のタスクのコストおよびそれぞれ割り当てられるイベントの量から全体コストを検出するステップ、を有する。

【 0 0 1 8 】

第2の側面によれば、本発明は、システム、特に複数の異なるタスクが設けられたコンピュータシステムの実時間を解析するための、コンピュータに組み込まれる方法であって

前記タスクは繰り返し実行され、

タスクの実行は、前記タスクの起動によりトリガーされ、これは前記タスクのイベントを示し、

1又は複数の記述的エレメントが前記イベントの時間相関をイベントストリームとして記述するために設けられ、

前記イベントストリームは前記イベントの最大時間密度および/又は最小時間密度を検出することができ、

少なくとも、イベントストリームの量が割り当てられ、少なくとも2つのイベントストリームによりキャプチャされるイベント全体の時間相関を記述するさらなる記述的エレメントを有し、

前記限定イベントストリームの少なくとも1つが少なくとも2つの限定イベントストリームに割り当てられ、それらはそれぞれ異なるタスクを起動し、

これらのタスクは少なくとも1つの結合グループに割り当てられ、

タスクには、それぞれ1つのイベントのための前記タスクによる前記リソースの一時的な使用をキャプチャする各コストが割り当てられ、

さらに、

前記タスクにより結合して要求される前記コストを決定するステップ、を有し、

該ステップは、

限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動す

10

20

30

40

50

る前記イベントストリームにより限定されるステップ、

前記割り当てられたイベントの量を考慮して前記限定イベントストリームの残りの結果を決定するステップ、

前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タスクに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タスクを起動する前記イベントストリームにより限定されるステップ、

前記ステップを前記限定イベントストリームの全てのイベントが割り当てられるまで繰り返すステップ、

個々のタスクのコストおよびそれぞれ割り当てられるイベントの量から全体コストを検出するステップ、を有する方法を提供する。

10

【0019】

本発明の両側面によれば、少なくとも2つのイベントストリームに、2つの異なる限定イベントストリームが割り当てられ、さらに、

前記タスクにより結合して要求される前記コストを決定するステップを有し、

該ステップは、

限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動する前記イベントストリームにより限定され、イベントの最小量はそれぞれ前記限定イベントストリームにより許容されるものであるステップ、

前記限定イベントストリームの残りのイベントを決定するステップであって、実際に割り当てられたイベントの量が考慮されるステップ、

20

前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タスクに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タスクを起動する前記イベントストリームにより限定され、他の限定イベントストリームが前記タスクに割り当てられるステップ、

前記ステップを前記限定イベントストリームの全てのイベントが割り当てられる、もしくはイベント量が前記グループの全てのタスクに割り当てられるまで繰り返すステップ、

個々のタスクのコストおよびそれらにそれぞれ割り当てられるイベントの量から全体コストを決定するステップ、を有する。

【0020】

30

前記限定イベントストリームは、時間インターバルにおけるイベントの密度を記述するポイントの量に割り当てられることが好ましく、前記限定イベントストリームは、イベントに対し異なるポイントの量を要求する少なくとも2つの限定イベントストリームに割り当てられる。

【0021】

第1のタスクの前記起動は第2のタスクにより実行されることができ、これらの起動の時間相関は、イベントストリーム（出力される前記第1のタスクのイベントストリーム、入力される前記第2のタスクのイベントストリーム）によってもキャプチャ可能である。

【0022】

少なくとも2つのリソースが前記システムに割り当てられることが好ましく、タスクの第1のグループが第1のリソースに割り当てられ、タスクの第2のグループが第2のリソースに割り当てられ、前記第1のグループの少なくとも2つのタスクがそれぞれ前記第2のグループのタスクを直接、もしくは複数のタスクを経由して起動する方法であって、

40

出力される前記第1のタスクのイベントストリームに対し、少なくとも限定イベントストリームを決定するステップ、

入力される前記第2のタスクのイベントストリームに対し、少なくとも第2のタスクに対する起動が第1のタスクにより直接実行されない限り、少なくとも限定イベントストリームを決定するステップ、

前記第2のタスクの、前記入力されるイベントストリームの前記限定イベントストリームを考慮して、前記第2のタスクにより結合的に要求されるコストを検出するステップ、

50

を有する。

【0023】

少なくとも1つの前記第2のタスクの前記限定イベントストリームを決定するために、前記第1のタスクの前記限定イベントストリームが使用される。

【0024】

前記タスクは、CPU又は電気回路モジュールにより実行可能なユニットであることが好ましい。

【0025】

一般的に、システムにおけるイベントストリームの量は、さらなる記述エレメント、好ましくは限定イベントストリームに割り当てられる。この限定イベントストリームにより、限定ストリームに割り当てられた全ての限定されたイベントストリームの互いに相関のある要求が記述される。例えばインターバル10msに対し最大量である5つのイベントストリームが、イベントストリームES__AおよびES__Bが割り当てられた限定イベントストリームES__ABに割り当てられ、最大量とはイベントの最大量を示す場合、これは実際のシステムにおいて、ES__AおよびES__Bの下層のシミュレーションが、10msのインターバルにおいて、5イベント以上を統合できないことを示している。このことは、例えば次のような場合に特に顕著である。任意の10msにおけるES__Aに6つのイベントが割り当てられ、任意の(別の)10msにおけるES__Bに7つのイベントが割り当てられ、すなわち1インターバルにおけるES__AおよびES__Bの合計が限定イベントストリームES__ABの値を超えている場合である。限定イベントストリームは、そのようなインターバルが少なくとも1つ存在する場合にのみ有利である。もしいずれのインターバルにおいても、同一インターバルにおいて限定イベントストリームに割り当てられるイベントの合計が限定イベントストリームの値を超えない場合、この限定イベントストリームは、解析に対して効果を有しない。

【0026】

最大イベントストリームに対する限定イベントストリームに対応して、最小イベントストリームに対する限定イベントストリームが設定されることがある。最小限定イベントストリームは、少なくとも1つのインターバルにおいて割り当てられるイベントストリームの数を超える場合のみ有利である。

【0027】

限定イベントストリームにより記述される依存性の一例として、2つもしくはそれ以上のタスク、例えば結合実行リソースの間の競合から得られるものがある。各タスク単体について、タスクから出力されるイベントの最高可能密度、すなわちタスクが割り当てられたインターバルに対し、タスクから出力されるイベントストリームの最大値は、スケジューリング方法の結合実行リソースの最高可能比率を生成する。しかしながら、第2のタスクに結合実行リソースのそのような高い比率が割り当てられないことがよくある。これは第2のタスクの出力されたイベントストリームの最大ではない値がこれらのインターバルを発生させていることを意味している。こうして、出力された第1のタスクのイベントストリームおよび出力された第2のタスクのイベントストリームの最大値が一致しないことがよくある。この関係は、競合状態の観点から、個々に出力されたイベントストリームの最大の可能な一致度を記述する限定イベントストリームにより記述される。

【0028】

限定イベントストリームは、時間インターバルに代えてイベント量やエネルギー量に合わせて設定されてもよい。これは、限定イベントストリームおよび割り当てられた限定イベントストリームは同種の値に基づくものにあるという点で重要である。

【0029】

限定イベントストリームにおいて、例えば重み係数等の点の量は時間インターバルに割り当てられることがあり、その場合、どれだけの点が当該限定イベントストリームの1以上のイベントに費やされる必要があるかは限定イベントストリームの観点から決定される。費やされる点の量は、限定イベントストリームに割り当てられる異なる限定イベントス

10

20

30

40

50

トリームに対して異なる値になる。こうして、例えば異なる限定イベントストリームの別々のランク付けや、イベント毎の変数複雑度のモデル化が可能になる。

【0030】

有利な実施形態においては、前述のメカニズムは特定の依存タイプに適用される。ここで、タスクの第1のグループは第1のリソースに割り当てられ、タスクの第2のグループは第2のリソースに割り当てられ、ここで第1のグループの少なくとも2つのタスクが、第2のグループの第1のタスクを、直接又はいくつかのタスク経由で、それぞれ起動する。以下のステップが必要とされる。

第1のタスクの出力されたイベントストリームに対する限定イベントストリームを少なくとも決定するステップ。

10

少なくとも第2のタスクに対する起動が直接行われない場合に、第2のタスクの入力されるイベントストリームに対する限定ストリームを少なくとも決定するステップ。

第2のタスクの入力されるイベントストリームの限定イベントストリームを考慮して第2のタスクの、結合所要コストを検出するステップ。

【0031】

限定イベントストリームは、依存性を考慮した解析方法により自動的に検出することが可能であり、リアルタイム解析を実行するユーザにより設定されてもよい。

【0032】

タスクが代替的にのみ起動可能な場合、限定イベントストリームにより記述可能な依存性は、別の種類のものである。これは、例えば所定のデータや、データの処理やさまざまなアクチュエータの制御を考慮した、代替物の選択を必要とする処理結果に応じて発生する。

20

【0033】

このような代替物の他の例として、エンジン制御ユニットがある。ここでは、例えば次のイグニッションポイントを適切に計算するために、さまざまな速度レンジにおいて、他のタスクが起動される。

【0034】

これら代替のタスクのために限定イベントストリームを使用することで、実時間解析の全てのステップにおいて、ただ1つの代替物、例えば各場合における最も複雑なものが確実に考慮される。

30

【0035】

詳細な記述が以下に示される。

【図面の簡単な説明】

【0036】

【図1】いくつかの異なる依存性を有する分散システムの例を示す。

【図2】競合する依存性を記述する限定イベントストリームの例を示す。

【図3】競合するタスクに対する限定イベントストリームのインターバルの計算を示す。

【図4】限定イベントストリームを用いた、最悪条件における応答時間に対する優先タスクの寄与の計算を示す。

【図5】イベントストリーム F 、 H 、および J のインターバルの改善のパーセンテージを示す。

40

【発明を実施するための形態】

【0037】

世代が進むにつれて、組み込みシステムに対する要求は高まっている。例えば車は、世代を追うごとにより多くのソフトウェア機能を有し、その結果、組み込みシステムはより複雑化している。新型の車は、いくつかのバスで接続された70個に達するECUを有している。エンジンマネジメントシステムやエンジンシステム等、多くの機能は実時間制約が課せられている。

【0038】

産業用ソフトウェア開発者により広く受け入れられた実時間解析において、タスクの最

50

悪条件での応答時間に対する厳しい制約は重要なものである。連関するタスクセットの依存性を考慮することは、優れたアプローチである。本発明は、実時間解析において、異なるタイプの依存性を統合する新たな全体論的モデルを提供する。本発明に係る一般的なモデルは、優先度の固定されたシステムのスケジューラビリティ解析に統合することが可能である。

【0039】

図1に、典型的な分散システムを示す。このシステムは2つのCPUと、1つのBUSを有する。この例では、各リソースに対する優先度スケジューリングが固定されているとする。このシステムは、プロセッサにおいて実行される8つのタスクおよびバスを有する。優先度は、図に示されるように割り当てられている。タスクの起動状態はイベントストリームで表現される。

10

【0040】

図1のシステムを解析するためには、各タスクの最悪条件における応答時間を計算する必要がある。その一般的な方法としては、システムの起動状態を記述する全てのタスクおよびイベントストリームが互いに独立であると仮定する方法がある。これは、イベントが最悪条件における応答時間解析の間に最大密度で発生することを意味する。なぜならシステム内の対応関係が考慮されていないためである。実時間解析の結果は、タスク間の干渉が常に最大と仮定したものであり、非常に厳しい結果が得られる。

【0041】

より厳しい応答時間制約を得るため、本発明では2つの依存性を導入する。第1の依存性は、同一のコンポーネントにより実行されるタスクが、該コンポーネントにおいて競合する状況を記述する、競合ベース依存性である。このような競合は、ここで示される状況のように、これらのタスクが互いに独立でないと仮定された場合に、特定のイベントが同一の密度では発生しないという影響がある。例えば、図1の G と H がこれに相当する。

20

【0042】

第2の依存性が、異なるイベントストリームからのイベントは、互いにシフトされた時間において発生しなければならないことを記述するオフセットベース依存性である。例えばもし、図1のイベントストリーム B と C について考慮する場合、イベントストリーム間で相関が存在すると仮定される。この依存性を導入する目的は、本発明に係るアプローチの汎用性を示すことである。

30

【0043】

導入されるこれら2つの依存性は、実時間解析に対するより厳しい制約を導き出す。そしてこれら2つの両者が組み込まれることが望ましい。従来技術では、タスク間の依存性の記述のための汎用アプローチとしての全体論的モデルは存在しなかった。

【0044】

以下に、後に議論される実時間解析に必要なモデルについて記述する。

【0045】

タスクモデル： τ は1つのリソース $R = \{r_1, \dots, r_n\}$ におけるタスクの a である。タスクは $\tau = \{c^+, c^-, p, \sigma\}$ という4つの要素からなる集合である。 c^+ は最悪条件での実行時間であり、 c^- は最良条件における実行時間であり、 p はスケジューリングの優先度（数が小さくなれば優先度は高くなる）であり、 σ はイベントストリームによるタスクの起動を示す。 σ_{ij} をタスク τ_i を実行する j 番目のジョブとする。

40

【0046】

タスクの各ジョブはその実行の最後にイベントを生成し、他のタスクに対する通知を行うと仮定する。

【0047】

イベントストリームモデル：イベントストリームモデルは、イベント制約機能に対し、有効な汎用ノーテーションを生成する。

【0048】

50

本発明においては、イベント制約機能 (t, θ) は全てのインターバル t に対して、長さ t の任意のインターバルのイベントストリームから生じるイベントの数の上界を生成する。このように、イベント制約機能は準追加的な機能であり、これはインターバル t, t' のそれぞれに対し、以下の条件 (1) を適用することを意味する。

$$(t + t', \theta) = (t, \theta) + (t', \theta) \quad (1)$$

【0049】

$(t, \theta), (t', \theta)$ は、任意の t もしくは t' のなかで可能なイベント数の最大値を返す。 $t + t'$ におけるイベントは、 t もしくは t' において発生しなくてはならない。従ってこの条件は成立する。

【0050】

本発明においては、イベントストリームはイベントエレメントの集合とも定義される。イベントエレメントのそれぞれは期間 p とオフセット a により与えられる (p, a) 。

【0051】

実際のシステムにおいて最悪条件でのイベントの密度が不明である場合、密度の上界をイベントストリームの記述に使用できる。そして任意のイベントシーケンスのモデル化が可能である。そして準追加的条件が成立するイベントシーケンスのみが、有効なイベントストリームである。

【0052】

イベントシーケンスおよびインターバル t のイベント制約機能は以下のように示される。

$$\eta(\Delta t, \Theta) = \sum_{\substack{\theta \in \Theta \\ \Delta t \geq a_\theta}} \left\lceil \frac{\Delta t - a_\theta}{p_\theta} \right\rceil \quad (2)$$

【0053】

逆関数として以下のインターバル関数が定義され、これによりイベントの数およびイベントストリームに対し、これらのイベントが発生可能な最小インターバルが与えられる。

【0054】

イベント数および t に対するインターバル関数は以下のように与えられる。

$$t^+(n, \theta) = \min \{ t \mid (t, \theta) = n \} \quad (3)$$

【0055】

イベントストリームのいくつかの例が、“Karsten Albers, Frank Bodmann, and Frank Slomka: Hierarchical event streams and event dependency graphs: A new computational model for embedded real-time systems. In ECRTS '06: Proceedings of the 18th Euromicro Conference on Real-Time Systems, pages 97-106, Washington, DC, USA, 2006. IEEE Computer Society” に示されている。(ここで参照することで包含されるとする。)

【0056】

既に議論されてきた組み込み実時間システムを拡張するために、限定イベントストリームは以下のとおり記述される。

【0057】

本発明においては、限定イベントストリームは、イベントストリームの集合に対しイベントの最大発生数を定義するイベントストリームと定義される。限定イベントストリームは、

$$\bar{\Theta} = (\Theta, \bar{\Theta})$$

と定義される。 $\bar{\Theta}$ は限定イベントストリームを記述し、

$$\bar{\Theta}$$

10

20

30

40

は限定イベントストリームが成立するイベントストリームの集合を示す。限定イベントストリームは以下の条件を満たす。

$$\eta(\Delta t, \Theta) \leq \sum_{\Theta_i \in \bar{\Theta}} \eta(\Delta t, \Theta_i)$$

【 0 0 5 8 】

例 1：イベントストリーム間に相関がなければ、

$$\bar{\Theta} = (\cup_{\Theta_i \in \bar{\Theta}} \Theta_i, \bar{\Theta})$$

と定義される。

10

【 0 0 5 9 】

例 2：図 1 は、限定イベントストリームの例を示す。 $\Theta_B = \Theta_C = \{(20, 0)\}$ 、およびこれら 2 つのイベントストリーム間のオフセットを 10 t.u と仮定する。イベントの累積的発生はイベントストリーム

$$\bar{\Theta} = (\{(20, 0), (20, 10)\}, \{\Theta_B, \Theta_C\})$$

で記述される。イベントストリームが互いに独立と見なされれば、インターバル $t = 5$ で 2 つのイベントが得られる。しかし限定イベントストリームは、インターバル t においていくつの累積的イベントが発生しうるかを記述している。この依存性により、インターバル t において 1 つのイベントしか得られない。

20

【 0 0 6 0 】

次に、限定イベントストリームがいかにして算出されるかを記述する。

【 0 0 6 1 】

本発明においては、 Θ ： t n は、所与のイベントストリーム

$$\bar{\Theta} := \{\Theta_1, \dots, \Theta_n\}$$

の関係から得られる、依存関係のある所与の数のイベントから、最小時間インターバルを割り当てる限定インターバル関数と定義される。そして限定イベントストリーム

$$\bar{\Theta}$$

30

は、次式で決定される。

$$\bar{\Theta} := v(\bar{\Theta}, \Delta\beta(n))$$

【 0 0 6 2 】

なお、

$$v(\bar{\Theta}, \Delta\beta(n))$$

および $\Delta\beta(n)$ は、依存性のタイプに応じて具体的に数式化される抽象的数式である。

【 0 0 6 3 】

以下において、本発明に係る競合ベースの依存性が記述される。図 1 に、タスク間のこの種の依存性が例示的に記述される。タスク τ_4 および τ_5 は同一のリソースにより実行される。これは、これらが当該リソースにおいて競合することを意味する。タスク τ_4 もしくは τ_5 の解析において行われる関連処理において、外部イベントストリーム τ_6 および τ_8 は独立して考慮される。

40

【 0 0 6 4 】

図 2 は、競合依存性を記述する限定イベントストリームの例を示す。時間軸の上の矢印は入力されるイベントを示す。時間軸の下矢印はタスクにより生成されるイベントを示す。ガントチャートのパート 1 では、競合しないタスクの場合が考慮されている。タスクの第 1 のジョブは、2 つの外部イベントがほぼ同時に発生しうる形でスケジュールされている。次のイベントは第 1 ジョブの第 1 イベントの後、可能な限り早く生成される。独立

50

である場合、次の２つのイベントは、ほぼ同時に発生する。しかしこれは、 τ_4 および τ_5 が同一のプロセッサにより実行されるためにジョブはタスク毎に実行されなければならないため、不可能である。これは下側のガントチャートに示されており、これはイベントの正確な発生を記述している。タスクの干渉があるため、外部イベントストリームの考慮を互いに独立に考慮したのでは不十分である。この干渉は限定イベントストリームによりモデル化が可能である。

【 0 0 6 5 】

図 2 に示されるように、２つの累積的外部イベントは同時に生成されうる。これは、より高い優先度を有するタスクは、第 2 のタスクに対し、その終了の直前に干渉するという事実に基づいている。その結果、２つのイベントはほぼ同時に発生する。これは、 n 個のイベントが同時に発生しうる n 個のタスクについてもあてはまる。

【 0 0 6 6 】

１個のタスクに対し、少なくとも $(n - 1) \cdot c^-$ 個の実行要求が n 個のイベントを生成するために実行される必要があると結論づけることができる。

【 0 0 6 7 】

限定イベントストリームを算出するために、 n 個のイベント間の最小距離が、競合ベースの依存性について限定インターバルを定式化することによって決定される。

【 0 0 6 8 】

R が、同一のプロセッサを共有する m 個のタスクの部分集合として規定され、

$$N = \{(n_1, \dots, n_m) : \sum_{i=1}^m n_i = n\}$$

を n 個のイベントの分散の集合として規定する。ここで各タスク i が n_i 個のイベントを生成し、限定インターバル関数は以下のように得られる。

$$\Delta\beta(n) = \min_{(n_1, \dots, n_m) \in N} \left(\max_{i=1, \dots, m} \left(\Delta\tau^+(n_i, \Theta\tau_i) \right), \left(\sum_{i=1}^m (n_i - 1) \cdot c_{\tau_i}^- \right) \right) \quad (4)$$

【 0 0 6 9 】

これは、 n 個のイベントがより小さい距離で発生しうると仮定されることから証明される。このことは、最小の組み合わせの 1 つが、より短い距離の結果を生じさせることを意味する。その結果、インターバル関数 $\Delta\tau^+(n_i, \tau_i)$ もしくは

$$\sum_{i=1}^m (n_i - 1) \cdot c_{\tau_i}^-$$

は、より短い距離を生成する。インターバル関数 $\Delta\tau^+(n_i, \tau_i)$ がより短い距離を生成すると仮定すると、その結果、イベントがイベントストリームにおける定義よりも短い距離で発生する。しかしこれはイベントストリームの定義と矛盾する。従って、最良ケースの実行時間総計は、より短距離で発生しなければならない。このような状況は、考慮されている実行時間の 1 つが仮定に含まれるものより短い場合にのみ発生しうるが、これは、既に全てのタスクについて最良ケースの実行時間が規定されていることと矛盾する。

【 0 0 7 0 】

本発明では、 n 個のイベントに対し、それらが発生する最小インターバルを提供する処理を導入する。この処理が図 3 に示される。本発明では、 $\Delta\tau^+(n, \tau)$ を効率的に算出するためにイベントストリームの正規化が導入される。

【 0 0 7 1 】

図 3 において、アウターループが上の最小オペレーション（４行目から 10 行目）により考慮される全ての組み合わせに渡って繰り返される。５行目は、各イベントストリームの全てのインターバルを、 $\max(\Delta\tau^+(n_i, \tau_i))$ により実行されたと見なす。内部ループ（７行目から 9 行目）は、

10

20

30

40

50

$$\sum_{i=1}^m (n_i - 1) \cdot c_{\tau_i}^-$$

といった最良ケース実行時間により生成される最小距離を算出する。最後に、インターバルの最小値が決定され、 n 個の累積的イベントが発生しうる最小インターバルが返される (10 行目)。

【0072】

本発明に係るアプローチの汎用性を示すために、Rodolfo Pellizzoni and Giuseppe Lipari (Improved schedulability analysis of real-time transactions with earliest deadline scheduling. In RTAS '05: Proceeding of the 11th IEEE Real Time on Embedded Technology and Applications Symposium, pages 66-75, Washington, DC, USA, 2005. IEEE Computer Society; (ここで参照することで包含されとする)) による遷移モデルにおいて発生するオフセットに関する問題に適用する。このアプローチは動的オフセットもカバーするものであるが、ここでは例としてのタスクシュミレーション間の静的オフセットのみにについて考慮する。

10

【0073】

本発明において、オフセット a を有する 2 つの厳密な周期的タスク τ_1 および τ_2 について、イベント τ_1 および τ_2 の間の最小距離 a' のみを算出するものとする。この最小距離は、タスクのピリオドの最小公約数 $x = \text{gcd}(p_1, p_2)$ を用いて、 $a' = \min(\text{mod}(a, x), \text{mod}(-a, x))$ により算出される。そして以下に示す

20

$$(n) = \min(t^+(n, \{(p_1, 0), (p_2, a)\}), t^+(n, \{(p_1, a), (p_2, 0)\})) \quad (5)$$

【0074】

ここで、イベントストリームを

$$\nu(\bar{\Theta}, \Delta\beta(n))$$

経由で直接設定できる。 $\text{mod}(a, x) = \text{mod}(-a, x)$ である場合、限定イベントストリームは

$$\bar{\Theta} = (\{(p_{\tau_1}, 0), (p_{\tau_2}, a')\}, \{\Theta_{\tau_1}, \Theta_{\tau_2}\})$$

30

である。 $\text{mod}(-a, x) < \text{mod}(a, x)$ である場合、限定イベントストリームは

$$\bar{\Theta} = (\{(p_{\tau_1}, a'), (p_{\tau_2}, 0)\}, \{\Theta_{\tau_1}, \Theta_{\tau_2}\})$$

である。タスクが 2 つ以上である場合にも、このアプローチは限定インターバル関数を計算するのに適用できる。

【0075】

限定イベントストリームを使用するためには、実時間解析、特に最悪ケース応答時間解析に対し、新しい概念を適用する必要がある。限定イベントストリームを考慮する場合、インターバル t におけるタスクの最悪ケースのコントリビューションの大きさを決定しなければならない。

40

【0076】

インターバル t におけるタスクの最大コントリビューションは、最大の最悪ケースにおけるタスクの実行が可能な限り発生した場合に発生し、そして限定イベントストリームまでの、最小の最悪ケースにおけるタスクの実行時間まで、可能な限り 2 番目に大きな実行時間を有するタスクは、他のさらなるタスクの発生を禁止すると仮定する。

【0077】

このことは、この仮定により与えられるものとは別のディストリビューションが存在すると仮定することで証明されうる。従って、この仮定によるパターンに従わないイベントが少なくとも 1 つ存在する必要がある。タスクのコントリビューションを増すために、イ

50

ベントは、最悪ケースの実行時間が仮定よりも大きいタスクをトリガーしなくてはならない。しかしこれは、最悪ケースの実行時間がより大きい全てのタスクに対し、既に呼び出しの最大数を仮定しているため、矛盾している。

【 0 0 7 8 】

応答時間解析は以下のように定義される。条件 $r^+(k, \tau) \leq d$ が成立する場合、タスクの集合は実行可能であって、実時間解析は成功する。イベントストリームを考慮した、タスクの最悪ケース応答時間は、以下のように算出される。

$$r^+(\tau) = \max_{k \in \mathbb{N}} \{ r^+(k, \tau) - \Delta t^+(k, \Theta_\tau) \mid r^+(k-1, \tau) > \Delta t^+(k, \Theta_\tau) \}$$

$$r^+(k, \tau) = \begin{cases} c_\tau^+ & k=0 \\ \min \{ \Delta t \mid \Delta t = k \cdot c_\tau^+ + \underbrace{\sum_{\tau' \in HP} \eta(\Delta t, \Theta_{\tau'}) \cdot c_{\tau'}^+}_{\text{calcHPStatic}} \} & k \geq 1 \end{cases} \quad (6)$$

10

【 0 0 7 9 】

優先度がより高いタスクにより生成される実行量は、イベント境界関数に最悪ケース実行時間に乗じることで算出される。不動点反復による方法により、最悪ケース応答時間は全てのジョブ k に対し算出できる。

20

【 0 0 8 0 】

$$\sum_{\tau' \in HP} \eta(\Delta t, \Theta_{\tau'}) \cdot c_{\tau'}^+$$

を組み込むために、図 4 のアルゴリズムが作成された。式 6 の残りの部分は変更されない。

【 0 0 8 1 】

このアルゴリズムは、考慮されているインターバル t 、解析中のタスクのジョブナンバー k 、検査対象であるタスク τ 、必要な限定イベントストリームの集合

$$\overline{\Theta}_{all}$$

30

、および τ より優先度が高い全てのタスクを有する HP_τ を、パラメータとして持っている。このアルゴリズムはタスクをそれらの最悪ケースの実行時間でソートし（ 8 行目）、全ての限定イベントストリームに対し、このストリームが t 内で許容するイベントの最大量を蓄積する（ 9 行目）。解析中のタスク τ の呼び出し数は、対応する限定イベントストリームから必ず減じられる（ 10 行目）。ループ（ 11 行目から 16 行目）では、優先度がより高い全てのタスクが考慮される。最悪ケースの実行時間が最も長いタスクが最初に考慮される。このアルゴリズムはタスクのイベントストリームにより該タスクに対する呼び出しの最大量を決定し（ 12 行目）、イベントストリームが存在すれば、その限界を決定する（ 13 行目）。これらの最小値は t 内でのタスクの最大コントリビューションを算出するのに使用される（ 14 行目）。第 2 のループ（ 15 行目から 16 行目）は、対応する限定イベントストリームを、使用されたイベントの分だけ減ずる。従って、これらのループはタスクに渡って、限定イベントストリームの量を分配する。この処理が、 t 内のより優先度の高いタスクの最悪ケースのコントリビューションを導き出す。

40

【 0 0 8 2 】

なお、応答時間解析の複雑さは依然、疑似多項的である。限定イベントストリームの算出の複雑さは、考慮されている依存性の種類による。競合ベースの依存性の問題の算出は、その複合的な複雑さにより困難なものになりうる。しかしながら解析は、この問題の影響を受けない。従って限定イベントストリームの上界を見出すことは、実行時間性能の向上を示唆するものである。

50

【 0 0 8 3 】

この新たなアプローチの意義が、以下の事例により示される。検査対象であるシステムは図 1 に示され、既に記述されている。表 1 はシステムのパラメータを示し、テーブル 2 はイベントストリームを示す。我々はこのシステムを、そのフォローが容易で、かつ新たな手順の全体を示されているため選択した。

CPU 1	τ_1	τ_2	τ_3	BUS 1	τ_4	τ_5
c^+	40	30	40	c^+	90	90
c^-	30	20	20	c^-	50	80
ϕ	1	2	3	ϕ	1	2
Θ	Θ_A	Θ_B	Θ_C	Θ	Θ_E	Θ_F

CPU 2	τ_6	τ_7	τ_8
c^+	50	35	35
c^-	30	25	25
ϕ	3	2	1
Θ	Θ_D	Θ_G	Θ_H

表 1 図 1 に示される分散システムのパラメータ

【 0 0 8 4 】

システムのイベントストリームを算出するために、本発明では“Steffen Kollmann, Karsten Albers, and Frank Slomka. Effects of simultaneous stimulation on the event stream densities of fixed-priority systems. In Spectra '08: Proceedings of the International Simulation Multi-Conference. IEEE, June 2008”(ここで参照することで包含されるとする)により示されるアプローチを使用してもよい。結果として得られるシステムのイベントストリームは表 2 に示される。これにより、依存性を用いて算出されたイベントストリームと、依存性を用いずに算出されたイベントストリームとの比較を行う。イベントストリーム Θ_B と Θ_C の間の静的オフセットを 100 t.u. と仮定する。

Θ	依存性を考慮	依存性を考慮せず
Θ_A	$\{(100,0)\}$	$\{(100,0)\}$
Θ_B	$\{(200,0)\}$	$\{(200,0)\}$
Θ_C	$\{(200,0)\}$	$\{(200,0)\}$
Θ_D	$\{(\infty,0), (100,90)\}$	$\{(\infty,0), (100,90)\}$
Θ_E	$\{(\infty,0), (200,150)\}$	$\{(\infty,0), (200,150)\}$
Θ_F	$\{(\infty,0), (200,140)\}$	$\{(\infty,0), (200,70)\}$
Θ_G	$\{(\infty,0), (200,100)\}$	$\{(\infty,0), (200,100)\}$
Θ_H	$\{(\infty,0), (\infty,80), (\infty,160), (200,310)\}$	$\{(\infty,0), (\infty,80), (\infty,160), (\infty,240), (200,370)\}$
Θ_I	$\{(\infty,0), (\infty,30), (\infty,60), (100,130)\}$	$\{(\infty,0), (\infty,30), (\infty,60), (\infty,90), (100,165)\}$
Θ_J	$\{(\infty,0), (200,55)\}$	$\{(\infty,0), (200,55)\}$
Θ_K	$\{(\infty,0), (\infty,70), (\infty,150), (200,300)\}$	$\{(\infty,0), (\infty,70), (\infty,150), (\infty,130), (200,360)\}$

表2 分散システムの全てのイベントストリーム。結果は、本アプローチの適用および非適用のうえで算出される。

【0085】

依存性の有る外部イベントストリームを決定するには、システムの限定イベントストリームを算出する必要がある。我々は2つのイベントストリーム

$\bar{\Theta}_1$

および

$\bar{\Theta}_2$

のみにについて考慮する。

$\bar{\Theta}_1$

は Θ_B と Θ_C との間のオフセットを記述する。

$\bar{\Theta}_2$

は Θ_G と Θ_H との間の競合ベースの依存性を記述する。

$\bar{\Theta}$	Θ	$\bar{\Theta}$
$\bar{\Theta}_1$	$\{(200,0),(200,100)\}$	$\{\Theta_B, \Theta_C\}$
$\bar{\Theta}_2$	$\{(\infty,0),(\infty,0),(\infty,80),(\infty,130),(\infty,210),(200,300),(200,310)\}$	$\{\Theta_G, \Theta_H\}$

表3 算出された限定イベントストリームの結果

【0086】

イベントストリームの算出後、システム解析の改善点をより詳細に確認した。まず、いくつかのイベントストリームおよびシステムの密度の改善を確認した。これについては表4および図5に示す。

n	Θ_F	Θ'_F	改善	Θ_H	Θ'_H	改善	Θ_I	Θ'_I	改善
1	0	0	0%	0	0	0%	0	0	0%
2	140	70	50%	80	80	0%	30	30	0%
3	340	270	20.58%	160	160	0%	60	60	0%
4	540	470	12.96%	310	240	22.5%	130	90	30.7%
5	740	670	9.45%	510	370	27.45%	230	165	28.26%
6	940	870	7.44%	710	570	19.71%	330	265	19.69%
7	1140	1070	6.14%	910	770	15.38%	430	365	15.11%
8	1340	1270	5.22%	1110	970	12.61%	530	465	12.26%
9	1540	1470	4.54%	1310	1170	10.68%	630	565	10.31%
10	1740	1670	4.02%	1510	1370	9.27%	730	665	8.9%

表4 イベントストリーム Θ_F , Θ_H , および Θ_I に対するアプローチの改善を示す。 Θ は依存性を考慮したインターバルを示し、 Θ' は依存性を考慮しないインターバルを示す。改善は%で示す。

【0087】

依存性は、イベントストリームの密度に影響を有するのみならず、最悪ケースの応答時間にも直接的に影響を及ぼす。タスク τ_3 の最悪ケースの応答時間は、150 t.u. か

ら 80 t.u. に減少している。このことは、依存性を考慮した解析の結果は、この場合、依存性を考慮しない解析の結果に比較して 46.66% 厳しいものであることを意味している。タスク 6 は、依存性を考慮しない場合 255 t.u.、依存性を考慮した場合 205 t.u. の最悪ケースの応答時間を有し、すなわち最悪ケースの応答時間において 19.6% 減少している。

【0088】

この例は、依存性が実時間解析を改善することを示している。これにより我々は汎用的なアプローチにおいて、異なる依存性がいかに容易に組み合わせられるかを示した。

【0089】

本発明は、分散実時間システムにおけるタスク依存性に対する全体論的モデルを実現する可能性を提供する。新たなアプローチは、固定優先度システムに適用された。2種の依存性が記述され、これらが新たに定義された限定イベントストリームによりどのように記述されるか示された。これにより、新たな種類の依存性が導入された。その効果として、本発明は実時間解析から依存性の複雑さを取り除いた。

【0090】

本発明は、図面および上記記述により詳細に示され記述された。これらの説明および記述は、例示的なものもしくは一例であって、限定的なものではない。そして後述する請求項の範囲で、通常の技術による変更や改良が行われてもよいことが理解されるべきである。特に本発明は、上記および後述される別の実施形態の特徴の組み合わせが付されたさらなる実施例をも含む。

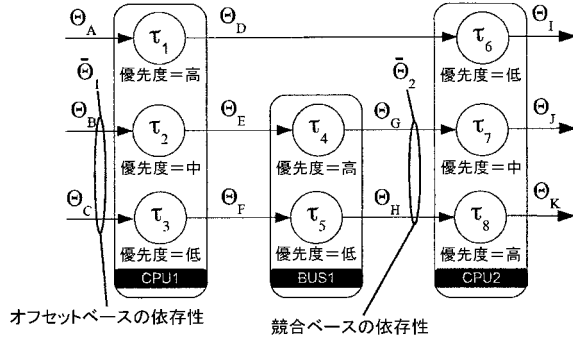
【0091】

さらに、請求項において「有する」という語は、他の要素やステップを除外するものではなく、英文請求項における不定冠詞「a」「an」は複数であることを除外するものではない。単一のユニットが、クレームで言及されている複数の特徴の機能を満たしていてもよい。また、英文請求項において特に属性や数値と組み合わせて使用される「essentially（実質的に）」「about（おおよそ）」「approximately（約）」等の語は、厳密なその属性、もしくは厳密なその値をそれぞれ含む。クレーム内のあらゆる参照符号は、発明の範囲を限定するものとして解釈されるべきではない。

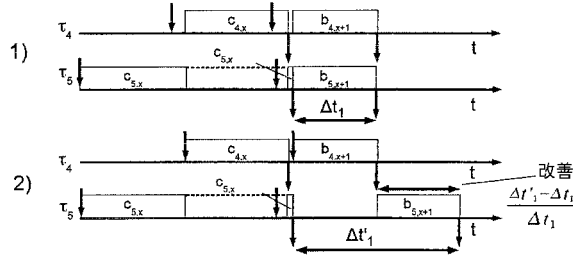
10

20

【図 1】



【図 2】



【図 3】

```

1 Δβ(n) {
2   ni ∈ ℕ;
3   Δeold = ∞;
4   for ((∀N|Σni=n)) {
5     Δt = max{Δt+(Θτ, ni)};
6     Δe = 0;
7     for (∀τi|τi ∈ ΓR) {
8       Δe = Δe + (ni - 1) · cτi+;
9       Δeold = min(Δeold, max(Δe, Δt));
10    }
11   }
12   return Δeold;

```

Fig. 3

【図 4】

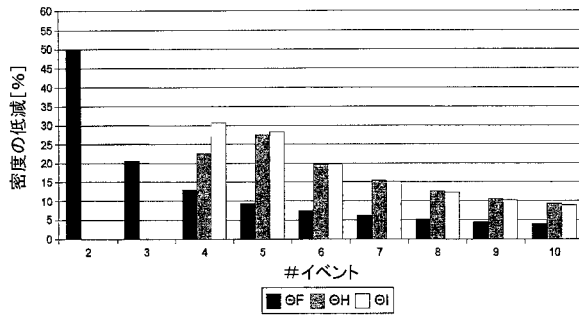
```

1 // Δt The interval which is considered
2 // k Number of calls of τ
3 // τ The task under analysis
4 // Θall All necessary limiting event streams
5 // ΓHP Set of τ: φτ > φτ
6 calcHPStatic(Δt, k, τ, Θall, ΓHP) {
7   Δe = 0;
8   sort(ΓHP); // ∀i ≤ j: cτi ≥ cτj;
9   (∀Θ ∈ Θall): η[Θ] = η(Δt, ΘΘ);
10  (∀Θ ∈ Θall|Θτ ∈ ΘΘ): η[Θ] = η[Θ] - k;
11  for (τj ∈ ΓHP) {
12    n = η(Δt, Θτj);
13    m = min(η[Θ]|Θτj ∈ ΘΘ);
14    Δe = Δe + min(n, m) · cτj+;
15    for ((∀Θ ∈ Θall|Θτj ∈ ΘΘ)) {
16      η[Θ] = η[Θ] - min(n, m);
17    }
18  }
19  return Δe;

```

Fig. 4

【図 5】



【手続補正書】

【提出日】平成22年12月14日(2010.12.14)

【手続補正 1】

【補正対象書類名】特許請求の範囲

【補正対象項目名】全文

【補正方法】変更

【補正の内容】

【特許請求の範囲】

【請求項 1】

システム、特にさまざまなタスクが設けられたコンピュータシステムの実時間を解析するための、コンピュータに組み込まれる方法であって

前記タスクは繰り返し実行され、

タスクの実行は、前記タスクの起動によりトリガーされ、これは前記タスクのイベントを示し、

複数の記述的エレメントが前記イベントの時間相関をイベントストリームとして記述するために設けられ、

前記イベントストリームは前記イベントの最大時間密度および / 又は最小時間密度を検出することができ、

少なくとも、イベントストリームの量が割り当てられ、少なくとも2つのイベントストリームによりキャプチャされるイベント全体の時間相関を記述するさらなる記述的エレメントを有することを特徴とする方法。

【請求項 2】

請求項 1 に記載の方法であって、

前記少なくとも1つのさらなる記述的エレメントが限定イベントストリームを記述することを特徴とする方法。

【請求項 3】

請求項 1 又は 2 に記載の方法であって、

前記さらなる記述的エレメントもしくは限定イベントストリームの少なくとも1つが、少なくとも2つのイベントストリームによりキャプチャされるイベント全体の最大時間相関を記述し、

少なくとも時間インターバル ($d t$) において前記限定イベントストリームにより許容されるイベントの量である時間インターバル ($d t$) のイベント量が、 $d t$ において非限定イベントストリームにより許容されるイベントの総量より少ないことを特徴とする方法。

【請求項 4】

請求項 1、2、3 のいずれかに記載の方法であって、

前記さらなる記述的なエレメント又は限定イベントストリームの少なくとも1つが、少なくとも2つのイベントストリームによりキャプチャされるイベント全体の最小時間相関を記述し、

少なくとも時間インターバル ($d t$) に対し、 $d t$ において前記限定イベントストリームにより許容されるイベントの量が、 $d t$ において前記非限定イベントストリームにより許容されるイベントの量を上回ることを特徴とする方法。

【請求項 5】

先に記載された請求項のいずれかに記載の方法であって、

前記システムには少なくとも1つのコンポーネント又はリソースが割り当てられ、前記コンポーネントの少なくとも1つには少なくとも2つのタスクが割り当てられることを特徴とする方法。

【請求項 6】

先に記載された請求項のいずれかに記載の方法であって、

少なくとも2つのタスクが1つのグループに含まれ、グループの前記タスクはリソース

に割り当てられ、前記リソースにおいて競合する前記タスクもしくはこのグループのタスクは、異なる時間においてのみ前記リソースにおける処理を行い、前記システムは前記リソースが前記グループの所定のタスクに割り当てられる時間点を決定する方法を有することを特徴とする方法。

【請求項 7】

先に記載された請求項のいずれかに記載の方法であって、
前記限定イベントストリームの少なくとも 1 つが少なくとも 2 つの限定イベントストリームに割り当てられ、それらはそれぞれ異なるタスクを起動し、
これらのタスクは少なくとも 1 つの結合グループに割り当てられ、
コストがそれぞれ、1 つのイベントのための前記タスクによる前記リソースの一時的な使用をキャプチャする前記タスクに割り当てられ、
さらに、
前記タスクにより結合して要求される前記コストを決定するステップ、を有し、
該ステップは、
限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動する前記イベントストリームにより限定されるステップ、
前記割り当てられたイベントの量を考慮して前記限定イベントストリームの残りの結果を決定するステップ、
前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タスクに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タスクを起動する前記イベントストリームにより限定されるステップ、
前記ステップを前記限定イベントストリームの全てのイベントが割り当てられるまで繰り返すステップ、
個々のタスクのコストおよびそれぞれ割り当てられるイベントの量から全体コストを検出するステップ、を有することを特徴とする方法。

【請求項 8】

請求項 1 に記載の方法であって、
前記限定イベントストリームの少なくとも 1 つが少なくとも 2 つの限定イベントストリームに割り当てられ、それらはそれぞれ異なるタスクを起動し、
これらのタスクは少なくとも 1 つの結合グループに割り当てられ、
タスクもはそれぞれ、前記タスクによる 1 イベントそれぞれに対するリソースの一時的な使用を検出する各コストに割り当てられ、
さらに、
前記タスクにより結合して要求される前記コストを決定するステップ、を有し、
該ステップは、
限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動する前記イベントストリームにより限定されるステップ、
前記割り当てられたイベントの量を考慮して前記限定イベントストリームの残りの結果を決定するステップ、
前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タスクに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タスクを起動する前記イベントストリームにより限定されるステップ、
前記ステップを前記限定イベントストリームの全てのイベントが割り当てられるまで繰り返すステップ、
個々のタスクのコストおよびそれぞれ割り当てられるイベントの量から全体コストを検出するステップ、を有することを特徴とする方法。

【請求項 9】

請求項 7 又は 8 に記載の方法であって、

少なくとも2つのイベントストリームに、2つの異なる限定イベントストリームが割り当てられ、さらに、

前記タスクにより結合して要求される前記コストを決定するステップを有し、
該ステップは、

限定イベントストリームの前記イベントを最も高いコストを有する前記タスクに割り当てるステップであって、ここで割り当てられる前記イベントの量が、前記タスクを起動する前記イベントストリームにより限定され、イベントの最小量はそれぞれ前記限定イベントストリームにより許容されるものであるステップ、

前記限定イベントストリームの残りのイベントを決定するステップであって、実際に割り当てられたイベント量が考慮されるステップ、

前記限定イベントストリームの前記残りのイベントを次に高いコストを有する前記タスクに割り当てるステップであって、次に、前記割り当てられるイベントの量が前記タスクを起動する前記イベントストリームにより限定され、他の限定イベントストリームが前記タスクに割り当てられるステップ、

前記ステップを前記限定イベントストリームの全てのイベントが割り当てられる、もしくはイベント量が前記グループの全てのタスクに割り当てられるまで繰り返すステップ、

個々のタスクのコストおよびそれらにそれぞれ割り当てられるイベントの量から全体コストを決定するステップ、を有することを特徴とする方法。

【請求項10】

先に記載された請求項のいずれかに記載の方法であって、

前記限定イベントストリームが、時間インターバルにおけるイベントの密度を記述するポイントの量に割り当てられ、前記限定イベントストリームは、イベントに対し異なるポイントの量を要求する少なくとも2つの限定イベントストリームに割り当てられる

ことを特徴とする方法。

【請求項11】

先に記載された請求項のいずれかに記載の方法であって、

第1のタスクの前記起動が第2のタスクにより実行されることができ、

これらの起動の時間相関は、イベントストリーム（出力される前記第1のタスクのイベントストリーム、入力される前記第2のタスクのイベントストリーム）によってもキャプチャ可能であることを特徴とする方法。

【請求項12】

先に記載された請求項のいずれかに記載の方法であって、

少なくとも2つのリソースが前記システムに割り当てられ、

タスクの第1のグループが第1のリソースに割り当てられ、タスクの第2のグループが第2のリソースに割り当てられ、

前記第1のグループの少なくとも2つのタスクがそれぞれ前記第2のグループのタスクを直接、もしくは複数のタスクを経由して起動する方法であって、

出力される前記第1のタスクのイベントストリームに対し、少なくとも限定イベントストリームを決定するステップ、

入力される前記第2のタスクのイベントストリームに対し、少なくとも第2のタスクに対する起動が第1のタスクにより直接実行されない限り、少なくとも限定イベントストリームを決定するステップ、

前記第2のタスクの、前記入力されるイベントストリームの前記限定イベントストリームを考慮して、前記第2のタスクにより結合的に要求されるコストを検出するステップ、を有することを特徴とする方法。

【請求項13】

請求項12に記載の方法であって、少なくとも1つの前記第2のタスクの前記限定イベントストリームを決定するために、前記第1のタスクの前記限定イベントストリームが使用されることを特徴とする方法。

【請求項14】

先に記載された請求項のいずれかに記載の方法であって、
前記タスクは、ＣＰＵ又は電気回路モジュールにより実行可能なユニットであることを
特徴とする方法。

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/051783

A. CLASSIFICATION OF SUBJECT MATTER INV. G06F9/48 ADD.		
According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) G06F		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practical, search terms used) EPO-Internal		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No
X	FRANK BODMANN ET AL: "Analyzing the Timing Characteristics of Task Activations" INDUSTRIAL EMBEDDED SYSTEMS, 2006. IES '06. INTERNATIONAL SYMPOSIUM ON, IEEE, PI, 1 October 2006 (2006-10-01), pages 1-8, XP031086712 ISBN: 978-0-7803-9759-0 cited in the application abstract page 1, left-hand column, paragraph 1 - page 3, left-hand column, paragraph 3 -----	1-14
<input type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principles or theory underlying the invention "X" document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art. "&" document member of the same patent family		
Date of the actual completion of the international search 15 June 2010		Date of mailing of the international search report 02/07/2010
Name and mailing address of the ISA/ European Patent Office, P.B. 5618 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016		Authorized officer de Man, Ronald

フロントページの続き

(81)指定国 AP(BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), EP(AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG), AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW

(72)発明者 コルマン シュテッフェン

ドイツ ウルム アルテ - ストラッセ 38 / 1

(72)発明者 スロムカ フランク

ドイツ ブラウシュタイン フェルディナント ザウアーブルッフ ストラッセ 16

Fターム(参考) 5B042 GA23 HH07 HH20 HH22 MC33