知能と複雑系 109－2
（1997. 8. 30）

知識に基づく相互関連タスクの協調制御機構

伊藤成記 渡邉豊英
名古屋大学大学院工学研究科
〒464-01 愛知県名古屋市千種区不老町
Phone: 052-789-2764
E-mail: {sitoh,watanabe}@watanabe.nuie.nagoya-u.ac.jp

あらまし

計算機の利用用途は単純な単一的なタスク処理から複合的なタスク処理に拡大している。複合的なタスクには様々な処理方法が考えられるが、それらを効率的に処理するためには相互に関連するタスクの扱いが重要になる。そこで、我々は様々な業務を効率良く遂行できる賢い人間の秘書をモデル化し、複合的な仕事を知的に、かつ効果的に処理する枠組を提案している。秘書システムは業務に関する様々な知識を知識ベースに持ち、抽象的な要求から処理の最小単位であるタスクを生成し、実行することでユーザの要求を実現する。本稿では特にタスクの類似性によるタスクの結合を提案し、タスク結合を有効的に利用するタスクの実行制御機構を報告する。

キーワード

秘書システム，タスク結合

Interrelated Task Management Mechanism Based on Knowledge Base

Shigeki ITOH and Toyohide WATANABE
Graduate School of Engineering, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-01, JAPAN
Phone: 052-789-2764
E-mail: {sitoh,watanabe}@watanabe.nuie.nagoya-u.ac.jp

Abstract

In this paper, we address a framework to manage various interrelated tasks, which are members of same complex jobs or composite units of different jobs, effectively. Such a task management is often observed in office environment. In order to satisfy such a task management we propose a task combination method, which regards several similar or related tasks as one executable job unit and evaluates it at once. As a framework that makes this task combination mechanism possible, we discuss a system model of three-layers process management mechanism such as event handler, task manager and task scheduler.

Keywords

secretary system, task combination

1 はじめに

計算機の利用用途は単純な単一タスク処理から複合的なタスク処理に拡大している。種々の関連するタスクを効率良く処理することが重要である。タスクのスケジューリング法は様々な分野で議論されている。従来の単一処理のみを扱う計算機システムでは利用者が複雑な要求を処理するために、利用者自身で各処理手続きの実行手順、各処理間における実行時の制約等を考慮して計画を立て制御する必要があり、また処理過程における計算機環境の変化、計算プロセスの実行状況などに対処しなければならない。例えば、人工知能における課題としても、多数のジョブのスケジューリングという形で議論されている [1]。我々はこのような複合的な処理を自動的に実行可能とし、知的かつ効果的なタスク実行制御機構を提案している [2]。本稿では、このような目標においてタスクの実行を効果的、効率的に実現する視点から、特に相互に関連し合うタスクの処理方法とそのために利用される知識構造について報告する。

2 タスク実行制御機構の概略

種々様々な要求から生成されるタスクは必ずしも総てが独立で、逐次的であることはない。相互に関連し合っている場合が少なくない。我々は複合的な処理を知的に、かつ柔軟に効率良く実行制御する枠組としてイベント・ハンドラ、タスク・マネージャ、タスク・スケジューラの3つの処理部から構成されるシステムを提案する。図1に、その構成イメージを示す。

イベント・ハンドラはシステムに発生する一連の要求 (要求イベント) に対応したワークを管理し、静的な順序制約を持つタスク集合であるアクションをタスク・マネージャに渡す。一方、タスク・マネージャは、アクションから具体的なタスクを生成し、これらのタスクの並列性、順序性を決定し、タスクをタスク・スケジューラに渡す。タスク・スケジューラはシステム内のタスクの実行状態に従って受け取ったタスクを次々に実行する。

ここで、イベント、タスク、アクション、オブジェクトなどは以下のように定義される。

[定義：イベント]
ユーザの要求を表すシステムへの入力
[定義：タスク]
処理内容を表す
[定義：オブジェクト]
タスクの実行主体
[定義：アクション]
タスクの生成グラフ、タスク集合と生成ルール
[定義：ワーク]
一連の仕事の手順、アクション集合と仕事の状態

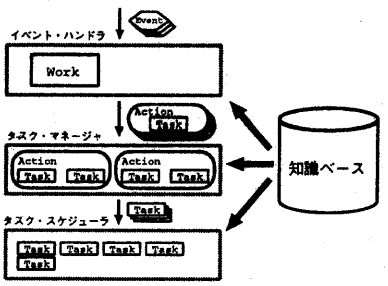

図 1: システムの概略

イベント・ハンドラはワークと呼ぶ一連の仕事のまとまりを管理するのに対して、タスク・マネージャは一度に実行できるアクションと呼ぶタスク集合を仕事の単位として管理する。また、これら2つの機構の静的な仕事の制御に対して、タスク・スケジューラはタスクの実行状態の管理という動的な視点よりタスクを実行制御する。

このような枠組でタスクの実行制御が実現される。タスクは処理の最小単位であり、これを実行するのがオブジェクトである。タスクの実行器であるオブジェクトから捉えると、タスクはメッセージとみなすことができる。従って、タスクにはそのパラメータとして、(1) タスクが送信され、実行されるオブジェクト名、(2) タスクの処理の種類を表すメソッド名、(3) タスクが実行時に参照するオブジェクト名リスト、(4) タスクを実行開始する時間帯が付帯している (図2)。

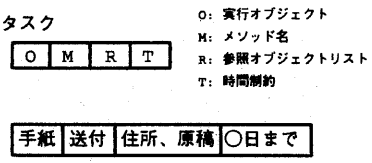

図 2: タスクのパラメータ

3 タスクの類似性とタスク結合

複数のタスクが実行可能な状態にあるとき、それらのタスクには少からず共通の処理手続きを含むタスクや、似かよった処理手続きから成るタスクが存在する。このように共通の処理手続きや、似かよった処理手続きが認められる二つ以上のタスクを類似関係にあるタスクと呼ぶ。類似関係にあるタスクを同時に実行することができるならば、システム全体の処理効率は向上する。

タスク間の類似関係は、タスクとそのパラメータを知識ベースに格納されるタスクの分類知識と類似関係知識を利用した検査法で判定される。

類似関係にあると判定された複数のタスクを結合して一つの結合タスクとしてまとめる処理をタスク結合と呼ぶ。結合タスクは通常のタスクと同様に扱われ、オブジェクトへと送信され、その処理結果は結合前のタスクの実行結果と同じである (図 3)。

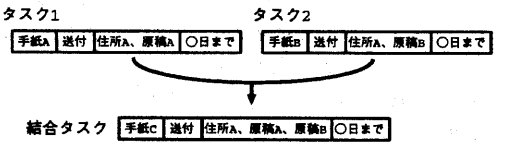

図 3: タスクの結合

システム全体の効率を向上させるには、より多くのタスクを結合することが必要となる。タスク結合の機会を増加させるための機構がタスク・マネージャであり、タスク結合はタスク・マネージャでを行われる。タスク・マネージャは、ワーク内のタスクを全て管理することで、実行が予定されている全てのタスクを把握し、タスク結合を予測する。

4 知識

計算機利用者の複雑な要求を実現するためには、その要求の実現のために利用する仕事の種類・手順の知識が必要である。多くの処理要求に確実に対応する手法として、種々様々な要求の全てに対応した知識をあらかじめ持つ方法が考えられる。この手法では、あらかじめ用意された要求に対応することができるが、様々な処理要求の全てに対する実現手順を知識として蓄積することはコストが高く、対応する処理数の増加を考えると実用的ではない。

そこでソフトウェア・モデリング、プログラミング方法論、システム構築法などの計算機技術の一つとして確率されてきたオブジェクト指向パラダイム [3, 4, 5] の is-a 関係を利用した知識の分類法を採用し、知識階層より無駄な知識の蓄積を避ける。また、タスクの結合に利用されるタスクの類似性を知識構造に取り入れ、インスタンス間の類似性導出関数を知識に組み込む。

管理される知識は、データをクラスに分類するための知識と、類似性のクラス間のデータを比較するための (類似知識) に分けられる。通常のオブジェクト指向パラダイムに利用されるクラス定義は、クラス名、属性名、メソッド名から構成され、それぞれのクラス間には is-a 関係を表すリンクが存在する。類似知識は、各クラス定義に存在し、そのクラスのインスタンス (オブジェクト) 間に存在する類似性を定義する。この類似性は 1 つのクラスに対して複数の視点から定義できる。例えば、名前、履修科目を学生クラスを考えると、ある学生ともう 1 人の学生の関係を表す類似関係には、同じ科目を履修する点での類似関係や、名前が類似する場合が考えられる。

4.1 知識表現

類似関係の定義を含むクラス階層を表現するために、クラス定義内のメソッドの宣言として類似関数を記述する。類似関数とは、引数として 2 つの同じクラスのインスタンスであるオブジェクトを取り、返り値として真理値を持つ。返り値が真ならばそれらのオブジェクトには類似性があり、偽であれば類

似性がない。

類似関数は記述法が他の関数と異なり、類似関数内で呼ぶことができる関数に制限される。類似関数内では、そのクラスが持つ属性の類似関数のみが利用でき、その結果の演算結果が類似関数の返り値となる。

例えば、学生クラスが属性として名前、履修科目を持つとすると、学生クラスの類似関数は次のように書ける。

```
class Student {
  Name name;
  Course course;
  Boolean byName(Student s1,s2) {
    return name(s1.name, s2.name);
  }
  Boolean byCourse(Student s1,s2) {
    return course(s1.course, s2.course);
  }
  Boolean student(Student s1,s2) {
    return name(s1.name, s2.name)
           & course(s1.course, s2.course);
  }
}
```

4.2 類似性の利用

類似性は、同じクラスから派生したサブクラスに属する2つのオブジェクトを引数とした類似関数の返り値によって判定できる。類似性は、複数の仕事を処理する環境において、似ている複数の仕事をまとめて処理することで、単体で処理するよりも効率的に等価な処理ができるという現実世界の人間の行動をモデル化するために利用される (図4)。人間における複数の仕事の類似関係の判断は類似関数に対応し、複数の仕事をまとめて実行することはタスク結合に対応する。タスク・マネージャで管理されるタスクに対して類似関数を呼び、その結果に従ってタスク結合が行われる。

5 知識の適用

仕事を遂行するために利用される知識は、処理対象となる事物を表す知識 (オブジェクト知識) と、処理手順などの規則を表す知識 (規則知識) の大きく2つに分類できる。

オブジェクト知識の中には、実際に扱われる事物の分類を表すクラス階層などのあらかじめ用意できるものと、処理対象の動的に変更される状態を把握するための記憶領域としての知識に分類されるが、

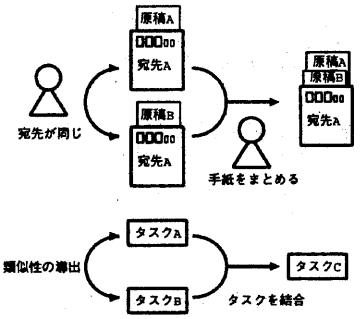

図 4: 仕事をまとめる

ここではクラス階層とクラス定義であらかじめ用意できる構造のみを議論する。

5.1 知識の構成

秘書システムにおいて知識は次のように構成される。

- オブジェクト知識
- 特性
- タスク
- イベント
- ワーク (状態遷移図)
- アクション (タスク生成グラフ)

オブジェクトは仕事の実行主体であり、タスクは仕事の内容を表す。

ワークは一連の仕事の状態を管理するために利用される。イベントはユーザの要求などのシステムへの入力であり、ワーク内に存在する状態遷移図の遷移を引き起こす。状態遷移によってタスクの集合であるアクションが生成される。アクションはタスクの生成知識 (タスク生成グラフ) でもあり、生成知識とオブジェクト知識からタスクを生成し、生成されたタスクを実行することで要求を実現する。

オブジェクト・クラスの定義は、オブジェクトが持つ特性と他のオブジェクトへの参照、タスクの定義、類似関数の定義から構成される。特性クラスは特性の集合と類似関数の定義から構成される。

オブジェクト知識が基本となり、オブジェクト知識を利用して他の知識を構成していく。図5はそれぞれの知識がどのように関連しているかを表す。

5.2 知識の利用例

オブジェクト

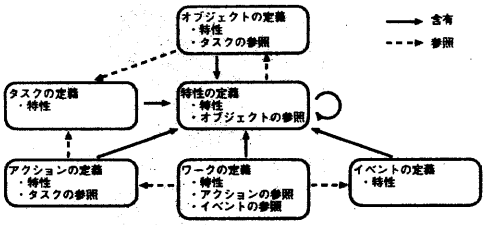

図 5: 知識の関連

図5のようにオブジェクト、特性、タスクの定義は互いに関わり合っている。オブジェクトはタスクの実行主体であり、システムから送信されたタスクに従って処理を行う。タスクを送信するためには、システムがそれぞれのオブジェクトに対して実行可能なタスクの種類を知識として持つ必要がある。システムはオブジェクトの性質に関する知識を特性の知識とタスクの知識を利用して表す。オブジェクトを表す適切なデータを特性として記述し、処理可能なタスクをオブジェクトの処理能力として知識に記述する。

また、知識としてのオブジェクトは、知識内のオブジェクト・クラスのインスタンスとして表され、実在するオブジェクトと対応する。図6に知識のオブジェクトと実在するオブジェクトの対応を表す。

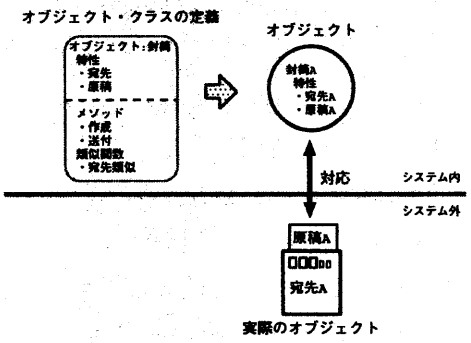

図 6: オブジェクト

タスクの条件指定

イベント、ワーク、アクションは、一連の仕事(ワーク)を遂行するために利用される。イベントには、ワークに与えられる引数が付帯しており、その引数から要求に関する情報を得る。ワークの引数は、アクション、タスクへと受け継がれるため、ワークの引数指定は間接的にタスクの引数指定となる。この情報には、特性の型、値による指定や、仕事を実行するオブジェクトの指定、オブジェクトの条件による範囲指定がある。条件による範囲指定は、オブジェクト・クラスのクラス名で指定するクラス指定と、そのクラスの特性の値を指定する特性指定を利用して記述される。範囲を持つ指定法を導入することで、システムが効率の良い実行オブジェクトを選択することができる。

6 タスク結合によるスケジューリング

タスクは生成から実行の過程で、実行待機、実行可能、実行中の3つの状態をとる。タスク結合は実行待機、または実行可能状態で行われる。しかし、従来の方法 [2] では、タスクは生成されると同時にタスク・スケジューラに渡され、実行順序制約のために待機状態となる場合を除き、常に実行可能状態となる。実行可能タスクはタスク・スケジューラ内の制約が解消されると同時に実行されるため、タスク結合のための類似性の検出のための時間が少ない。この機構では実行待機状態となるタスクが少なく、類似性の検出には不十分である。そこで、タスクの類似性をより多く検出するための機構としてタスク・マネージャが導入される。

6.1 タスク・マネージャ

タスク・マネージャは入力をアクション、出力をタスクとし、タスクの類似性に基づいたタスク結合を行う機構である。また、タスク・マネージャはアクション内で定義される順序関係に従ってタスクの実行順序を決定するだけでなく、タスクの実行状態を管理することでタスクの実行終了を確認し、実行順序制約を持つタスクの実行タイミングを決定する。

タスク・マネージャは、イベント・ハンドラで生成されたアクションからタスクを生成する。イベント・ハンドラはワーク内の総てのアクションをタスク・マネージャに送るため、タスク・マネージャは

ワーク内で生成される可能性のある総てのタスクの間でタスクの類似性を検出できる。(図 8)

タスク・マネージャはタスクのスケジューリングのために以下の手続きを行う。

1. タスクの生成

   アクションが入力されるとアクションと知識からタスクを生成する。

2. タスクの類似性検出

   生成されたタスクと類似性のあるタスクを調べ類似関係にあるタスクにリンクを張る。

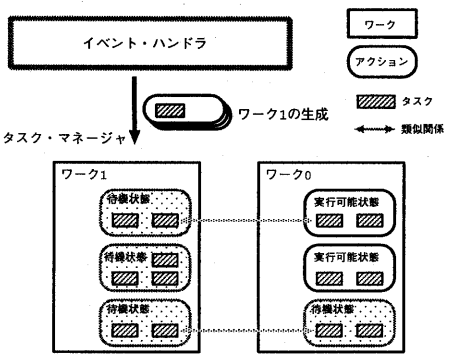

図 7: ワーク生成と類似性の判定

3. タスクの待機

   タスクに指定された実行可能な時間帯内で、期限内に実行可能なタスクは実行可能なタスクであっても待機させ、即時に実行しない。

   実行順序制約のあるタスクは、先行するタスクが終了するまで待機させる。

4. タスクの実行

   実行期限が近づいたタスクをタスク・スケジューラに渡す。結合できるタスク群は、そのタスク中の最も早い実行期限が近づいたときに、タスクを結合し、タスク・スケジューラに渡す。

7 おわりに

本稿では、複数のタスクを実行しなければならない環境で、相互に関連するタスクの検出し、全体の効率を向上させる試みを述べた。タスクの類似性に基づいたタスク結合と、その類似性を効率的に検出するためのタスク実行制御機構を示した。

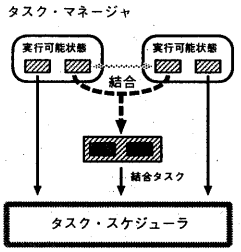

図 8: タスクと結合と実行

このような応用は人間の賢い秘書のように、複数の仕事を効率的に処理しなければならないオフィス環境の情報システムにとって重要な機構であり、秘書システムの実現に向けてシステムを開発する必要がある。

謝辞

日頃，熱心に討論していただいている佐川雄二講師，朝倉宏一助手をはじめ，渡邊研究室の皆様に感謝致します.

参考文献

[1] 宮下和雄: “スケジューリング問題へのアプローチ (1) : 人工知能にお ける課題”, 人工知能学会誌, Vol. 10, No. 6, pp. 880–887 (1995).

[2] 渡辺裕之, 渡邊豊英: “相互関連タスクの実時間処理のための連携的実行制御機構”, 情報処理学会論文誌, 第 37 巻, pp. 1215–1226 (1996).

[3] B. Meyer: “*Object-Oriented Construction*”, Prentice Hall (1988).

[4] J. Rambaugh: “*Object-Oriented Modeling and Design*”, Prentice Hall (1991).

[5] A. Hutt ed.: “*Object Analysis and Design : Description of Methods*”, John Wiley & Sons (1994).