1E-6

統合プログラミング環境 ORAGA (I)
並列オブジェクト指向言語 DinnerBell

°神田陽治 ・ 立川江介 ・ 田中英彦 ・ 元岡 達
東京大学 工学部

並列オブジェクト指向言語DinnerBellは、高い並列性を最大限に生かす言語を開発することを目指している。

オブジェクトは内部状態を持つ。オブジェクトの世界の計算は、メッセージの受信と引き続く手続きの実行によって、オブジェクトの内部状態が変化することで進む。オブジェクトは原則的に、相手を選ばずメッセージを受け取る。よって、メッセージの到着順は計算結果に重大であり、ときには計算結果を非決定的なものにまでする。理論の上ではともかく現実の並列実行環境では、発信されたオブジェクトが即時に相手に到着することは保証できない。大きな負荷なしには、全てのオブジェクトに共通な時間を設定することはできないからである。

一般的に言えば、計算結果が非決定的になってしまうのは、モジュールの分割の仕方が誤りなのであって、コントロール中心に分割をやり直すべきである。ところがオブジェクトは、現実に即したデータ中心の分割の結果得られたモジュールであるから再分割はできない。オブジェクトにどこまで安全に並列性を導入することができるかを検討し、並列オブジェクト指向言語の持つべき性質を明らかにする。

1. 並列オブジェクトの概念

関心があるのは、オブジェクトの枠組みにどこまで並列性を導入できるかを検討することである。そのためには、並列に動くオブジェクトの姿から考察を始めるべきだろう。 並列オブジェクトという言葉で、実際に並列に動き、互いに協調して活動できる「理想」のオブジェクトを指す。必要な性質は、

(1) 複数のメッセージ送受信と実行のオーバーラップ

同時に複数のメッセージを受け、できる限り多くの仕事を並行に分割処理し、複数のメッセージを送り返す能力を持つこと。仕事を頼んだ側は、仕事の終了を待つことなく自分の仕事を続け、仕事を進める側は自分の判断で仕事を進める。待ち合わせは、結果が戻っておらず、仕事をそれ以上進められない時に始めて起こる。すなわちデータフローに応じて挟みこまれる。

活動の単位がより細分されることで、仕事を依頼する指令も結果の返送も、複数のメッセージに分割して送れるようになる。特に並列実行下では結果が一つだけ返るとは限らなくなるので、複数の結果を返し、また受け取れる能力が必要である。

(2) データ抽象およびコントロール抽象モジュールとして

オブジェクトに高い並列性を導入することは、並列計算機での効率良い実行を期待させるが、オブジェクトの持つ性質を破壊してまで導入することは論外である。オブジェクト導入の目的は、ソフトウェアを作りやすくすることにあった。並列オブジェクトのプログラミングに多大の注意深さを必要とするのでは、オブジェクトを採用した意義が失なわれてしまう。

オブジェクトに際立つ性格は、自己完結性である。オブジェクトは自分にできることをデータ抽象して公開する。仕事を頼む側は決められたプロトコルを守って通信し、メッセージを介して指令を伝えるだけで良い。後は細かいことは気にせずに結果を待てばよいのである。並列性を入れる上においても、依頼側が指令を送るタイミングを注意深く計る必要があるのでは、すべての呼び出しで同じ小細工を繰り返さなければならない。並列オブジェクトが変更されたとき、すべての呼び出しを変更する羽目になる。並列オブジェクトはデータ抽象のみならず、コントロール抽象の単位としても自己完結すべきである。仕事を頼む側には、並列性を利用するしないの意識をさせたり、選択させたりするべきではない。仕事を依頼するプロトコルは、ただ一つに限るべきである。

2. 並列記述の方法

後の議論のために並列記述の方法をまとめる。一般に同期には大きく二つの意味があるが、同期、順序化に整理する。

並列記述には二つの面がある。仕事の分割と調節である。仕事を早く終わらせる有効な方法は、仕事を複数に分割することである。全体でまとまった結果を得るためには、ときどき相談し合っては、個々の仕事の開始や終了、共有物の使用など、進み具合いを調節する必要もある。実効ある仕事の遂行には、この二つの記述が不可欠であり、並列プログラミング言語はこれらを実現する計算機構を備える。注意すべきは、より強力な分割機構を持つ言語は、数多い活動を正しく調節するために、さらに強力な調節機構を備えなければならない、ことである。

(1) 仕事の分割としての宣言

多くのプログラミング言語では仕事の分割を、個々の仕事の開始点を陽に宣言させることで記述する。この宣言された分割点に仕事の実行が達した時に、仕事の分割が行われる。CoBeginブロック内の文や、単一代入変数を参照する式は、並列に実行されるべき仕事の開始を宣言している。Fork命令も活動の複製を作る位置を決める。

(2) 仕事の調節としての同期

調節記述の一つに同期がある。同期は相互排除に基づく調節機構である。共有資源への排他アクセス制御では、競争者がいなければ即座にアクセスが認められる。複数の者が同時にアクセスを試みるとき調停され、ただ一人のアクセスのみが許可される。リーダ・ライタの問題は代表的な同期問題であり、共有メモリセルの排他制御が必要である。

(3) 仕事の調節としての順序化

二つ目の調節記述は順序化である。順序化はあらかじめ決められた条件が満たされたことで活動を開始・再開・終了させる調

節機構であり、活動の生起を制御するときに利用する。サーバ・リクエスタの問題は代表的な順序化の問題である。バッファへの読み書きを順序付けて、二度読みや読み飛ばしが無いことを保証しなければならない。

3．DinnerBellの並列記述〔1〕

並列処理を宣言する方法としては、単一代入則を用いる。これで、オブジェクトの内部での並列活動数を高めることが可能になる。これらの活動の結果、多くのメッセージが送出され、オブジェクトの世界全体の活動数も著しく増大する。

オブジェクトの内部での並列活動の同期は、排他領域を設定することで取る。オブジェクトの内部での並列活動の順序化は、単一代入待ちにより、データフローに沿って行われる。複数のオブジェクトを排他資源に指定する方法は、目下のところ与えない。一応、管理用のオブジェクトを置き、排他領域を使うことで解決できるからである。

複数のオブジェクトが関係する順序化が、トピックである。オブジェクトの世界ではメッセージの到着で物事が起こるから、順序を制御することは、オブジェクトへのメッセージの到着順を制御する事に他ならない。そこで順序化は二つの場合に分けて考えればよい。メッセージの到着制御を望むものが、当のオブジェクトの意思である場合と、第三者のオブジェクトの意思の場合である。前者の順序化機構を協調、後者のそれを統御と呼ぶことにする。

協調とは、あるオブジェクトが別のオブジェクトにメッセージを送り、返事メッセージを受け取るときに、課せられるべきメッセージの到着順に関する約束事である。1(2)で述べた、仕事を依頼するときのプロトコルはただ一種に限るべきという主張は、協調の機構はただ一つに限るべきと言い換えられる。協調は、並列性を高く出せるように緩いものである一方、それで解決できないで残る非決定性を、統御で解決できる位に制限が強いものでなくてはならない。DinnerBellの協調は、メッセージ送信に関しては、プログラムの文面に書かれた順番に相手オブジェクトへ到着することを保証する。返事メッセージに関しては、返事の戻りアドレスにカラーを配することで、しかるべき活動に答えが返るように保証する。

統御とは、あるオブジェクトの意思で、他のオブジェクトの間のメッセージ送受の順序を制御可能にする方法である。例えば、日付印刷をしたいとき、年月日と時分秒のオブジェクトに同時に仕事を頼んで、結果を直接プリンタに出してもらう場合に、プリンタへのメッセージ伝達を年月日、時分秒の順に順序付ける必要は、仕事を依頼したオブジェクトの意思により生まれる。DinnerBellの統御は、セクレタリと呼ぶ、メッセージのマージ、蓄積、転送能力を持つ（組み込み）オブジェクトを、メッセージフローの要所に動的に嵌め込む事による。嵌め込みは、日付印刷をしようとするオブジェクトの実行の一部として行う。日付印刷オブジェクトは、プリンタへの実際への印刷とは無関係に仕事を終了することができる。セクレタリの役目は動的にメッセージパスを決めるという意味で、μ-calculus〔2〕での Conduitに似ている。動的にパイプをオブジェクト間に張ることであるとも見なせる。

4．チェスプレイの問題

並列オブジェクト指向言語の、順序化解決能力を試すための問題を設定する。これは同期問題を追及するために、一連の問題が提案されたことに習うものである。

> チェスプレイ問題　二人のチェスプレイヤーは、チェス盤を前に、白黒の駒を交互に動かす。白が先手、黒が後手。手番になったプレイヤーは、制限時間以内に最良手と判断した駒を動かすか、ゲームを投げる選択をする。審判は、手番のプレイヤーが時間制限を過ぎても駒を動かさないか、ゲームを投げたとき、あるいはチェス盤を常に判定して、引き分けを含めて勝敗がついたと判定したら、ゲーム終了を二人のプレイヤーへ宣告する。各プレイヤーはゲーム終了まで休むことなく、チェス盤上のその時点までの駒の動きを材料に、最良手を見つける努力をすることが望まれる。このゲームをプログラムせよ。

二人のプレイヤーオブジェクトが交互にメッセージを介して手を教え合う。局面の変化に対応して無駄な探索を中止する、賢いプレイヤーをシミュレートできるかどうかに関心がある。手番のプレイヤーオブジェクトは、相手に自分の手を伝えた後も、休むことなく局面を読み続けなくてはならず、同時に、いつ来るかわからない相手の手も待たなくてはならない。しかも手を受ければ直ちに無駄な探索は止めて、次の手の読みを始めなければならない。

審判オブジェクトは、二人のプレイヤーからのメッセージで打たれた手を知るものとする。（チェス盤オブジェクトが、二人のプレイヤーの間にある実装は採らない。）二人のプレイヤーは交互に手を伝えてくるが、審判に正しく交互に到着するような配慮が必要である。なぜなら、（出所のオブジェクトが異なる）複数のメッセージの到着順番は、並列環境下では一般的には保証できないからである。特に、二つのプレイヤーオブジェクトとは別のプロセッサへ、審判オブジェクトを割り当てたときでも、正しい順番でメッセージが審判へ到着することを保証しなければならない。

手にプレイヤー名を付け加えても解決にはならない。審判が遅れて来るはずの手が実際に来るまで、先に来た手を記録する解決案は、現実とは違う不自然な解決法である。

現在のDinnerBellの並列記述が、この問題を解けるかどうかを評価し、改訂強化する仕事はこれからである。

〔1〕情報処理学会　ソフトウェア基礎論　11- 3 (1984).
〔2〕ACM　JACM　27- 2 (1980).