

プロセスプランのペトリネット表現とそのスケジューリングへの応用

北海道大学工学研究科 ○水野陽太 金井理 岸浪建史

要旨

CPP-net[1]は、プロセスプランにおける代替タスクをペトリネットとして表現した動的なプロセスのモデルである。本研究では、このプロセスプランのためのペトリネット表現をスケジューリングへの入力として利用するために、時間、リソースインスタンス、プロダクトインスタンスの表現が可能な拡張CPP-netへ変換する方法を提案する。

1. はじめに

現在、図1に示すようなスケジューリングやジョブショップコントロールにおける製造プロセスのペトリネット表現が、広く研究されている。これに対し、プロセスプランニングを対象とした代替タスクを含むプロセスプランを表現するためのペトリネットとしてCPP-net(Compact Process Plan-net)[1]が提案されている。CPP-netをスケジューリングに利用できれば、動的なスケジューリングに有効であると考えられるが、CPP-netで記述される情報のみでは十分とは言えない。そこで本研究では、CPP-netをスケジューリング入力用のプロセス表現として利用できる拡張CPP-netへ変換する規則を新たに提案し、その有効性をシミュレーションツールにより検証する。

2. CPP-net

CPP-netとは、複雑なプロセスプランをモデル化し、分析するための表現技術として提案されたペトリネットの一種である。CPP-netでは、タスクとタスク間の複雑な関係をモデル化できる。図2(a)はIDEF3に基づいたタスク実行順序表現の例であり、そのCPP-net表現が図2(b)である。CPP-netでは、タスクをトランジションで表現し、単数部品を対象としているため各トランジションにIP(Input Place)とControl placeが入力となる。タスク間の先行関係、並列関係、代替タスク間関係、タスクグループ化は、図3の様に表現される。

このCPP-netをプロセスプランの表現として用いる利点は次の通りである。1) 実行可能モデルである。2) プロセスの進行状況を視覚的に確認できる。3) or表現が可能であるため、代替タスクを表現できる。4) 可達性を検証できるため、実行可能なプロセスプランをすべて検証できる。

しかし、CPP-netをスケジューリングの入力として直接利用するには、次の様な問題点がある。1) タスクを実行するリソースのインスタンスが表現されていない。2) タスク実行時間が表現されていない。3) 一種類のプロダクトの複数インスタンスまたは複数種のプロダクトのインスタンスを同じジョブフロアで製造するプロセスを表現できない。

3. 拡張CPP-net への変換方法

スケジューリングの入力としてペトリネットを用いることは、1) スケジューリング結果の実行可能性を再検証できる。2) リソースの状態に応じてスケジュールの代替案を持てる。3) 部分ネットを追加することで簡単なスケジュールの制約表現が可能であるという利点がある。そこで、本研究では、図4の流れに従いCPP-netをスケジューリングへ入力可能な拡張CPP-netへ変換する方法を提案する。すなわち、前報[2]で提案したタスクの表現に必要な情報を図5に示すが、CPP-netでは表現できないTime, Resource, Productの表現について拡張を行う。その手順を具体例(図2, 図6から図9)とともに示す。

(1) CPP-netによる単体部品のプロセスプランのペトリネット表現(図2)

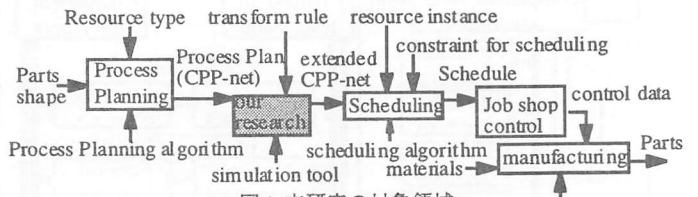


図1. 本研究の対象領域

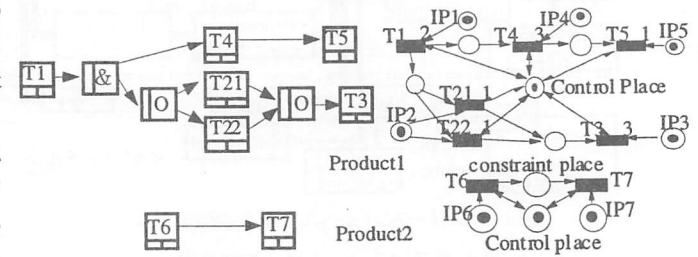


図2 単体部品のプロセスプラン表現 (CPP-net)

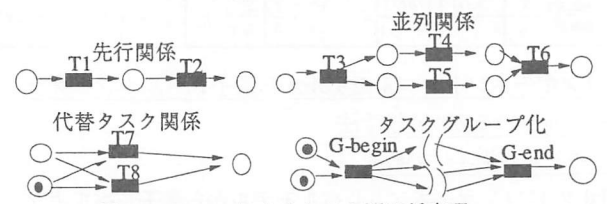


図3. CPP-netによるタスク間関係表現

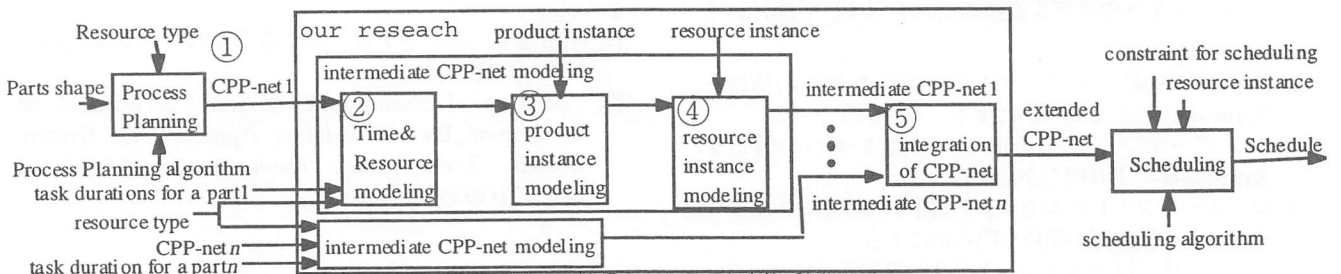


図4. 拡張CPP-net への変換手順

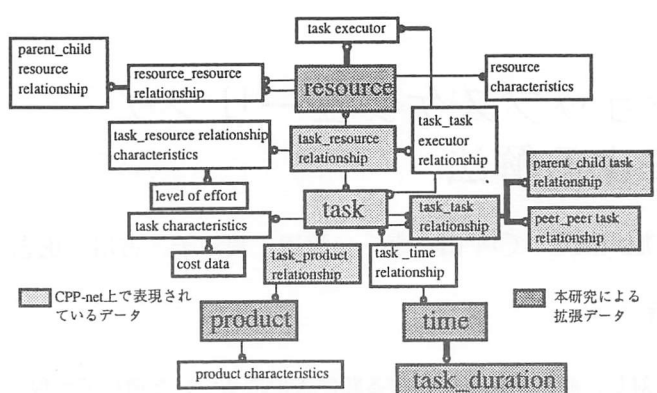


図5.タスク表現に必要なデータとCPP-net及び本手法の表現範囲

(2)リソース・時間情報を追加したペトリネット表現への変換 (図6)

(2-1)搬送, 保管タスクに対するトランジションの追加. 図7に示すようにプロセス内の各タスクは, 時間, プロダクト, リソース状態空間における直線として表現されるが, 加工タスクを連続的に行うためには, プロダクトの状態は変わらずに, 時間の状態が変化する搬送, 保管というタスクを追加する必要がある.

(2-2)トランジションの時間修飾化 (ただし, 待ちを表現する保管タスクに関しては時間0とする.)

(2-3)リソースタイプのプレースをアークによりトランジションに結合 (図6内のR1,R21,R3等のプレース)

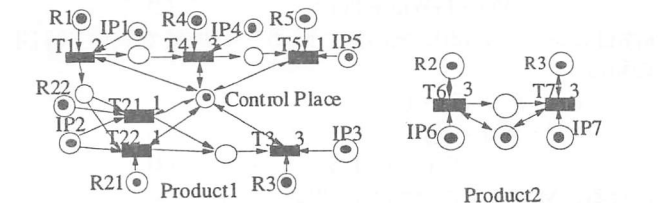


図6 時間・リソースタイプを考慮したペトリネット表現

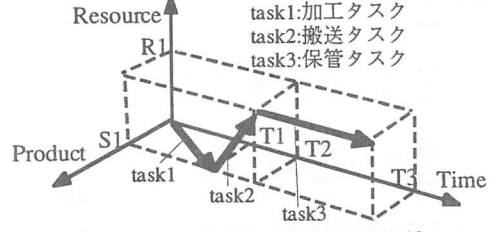


図7 ProductとResource,Timeの関係

(3)単一部品のインスタンスを考慮したペトリネット表現への変換 (図8)

- (3-1)最初のタスクのIPをsource Placeに変更
- (3-2)そのほかのIPを除去
- (3-3)並列タスクにNew control placeの追加
- (3-4)そのほかのcontrol placeを除去
- (3-5)終了を示すdummy task, sink placeの追加

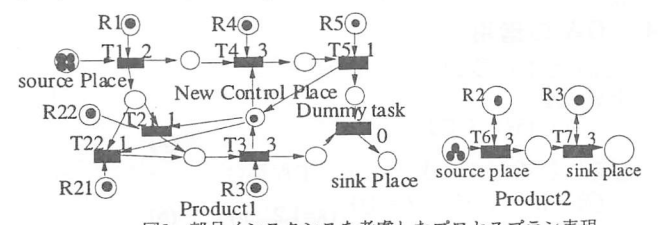


図8 部品インスタンスを考慮したプロセスプラン表現

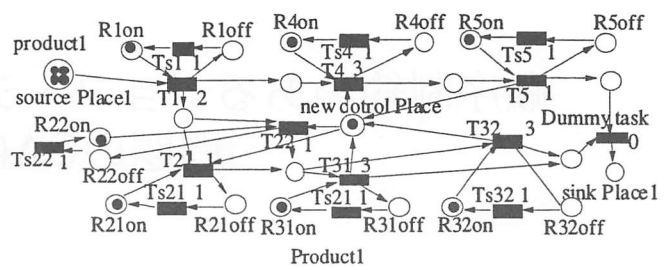


図9 リソースインスタンスを考慮したプロセスプラン表現

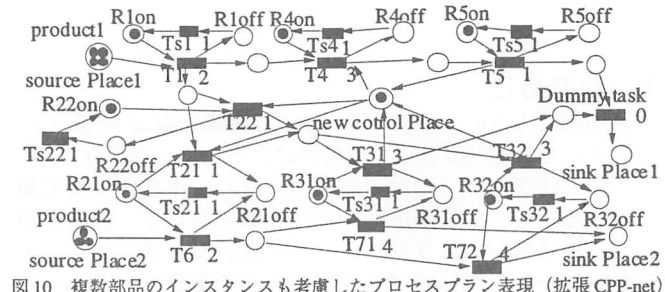


図10 複数部品のインスタンスも考慮したプロセスプラン表現 (拡張CPP-net)

(4)リソースのインスタンスを考慮したペトリネット表現への変換 (図9)

- (4-1)リソースタイプのプレースをリソースのonとoff状態遷移を表す2つのプレースと1つのトランジションに変更
- (4-2)タスクを実行可能なリソースインスタンスが複数ある場合にタスクトランジションを分割(T7→T71,T72等)
- (5)複数部品のプロセスを統合したペトリネット表現への変換 (図10)

各部品のCPP-netでの共通のリソースインスタンスに相当するプレースを共有するようにペトリネットを統合

4. 検証

シミュレーションツール (Visual Object Net++)で可達性の検証をした. その結果, 図6のCPP-netに対して, プロセスにおける部品の順送りの挙動が検証できた. また, 図6から変換された図9の拡張CPP-netに対しては, タスクT31とT71でリソースR31を競合しており, リソースが占有できない場合には, 空きリソースであるR32が自動的に選択され, 空きリソースを用いたタスクの効率的な実行が拡張CPP-netにより可能であることが確認された.

5. おわりに

プロセスプランのペトリネット表現から, 時間と複数部品, リソースのインスタンスといったスケジューリングの入力用として必要な情報を持ったプロセス表現である拡張CPP-netへの変換規則を提案し, その有効性を検証した.

参考文献

[1] D.Kiritisis, et al; Petrinet Representation for the Process Specification Language-Part1:ManufactureProcess Planning: <http://www.mel.nist.gov/psl/psl-secure>
 [2] 水野陽太他: 精密工学会春季大会学術講演会講演論文集 p.20(1998)