

通信システムにおける 設計仕様の統合方式の一考察

長谷川 晴朗 柴田 健次 湯山 さつき
沖電気工業(株)

1. はじめに

筆者らは、通信システムにおいて曖昧なユーザ要求から全体として整合のとれた設計仕様を作成するエキスパートシステム (EXPeRt system for an ESS) を構築中である。⁽¹⁾

本論文では、ペトリネットを用いてビジュアルに表現した設計仕様に対して、関数行列を使用した観点から解析を行ったので報告する。

2. サービスグラフの統合

EXPRESS における設計仕様の作成過程を図1に示す。まず通信システムに対する個々の断片的な要求(自然言語)は、Prolog表現された要求仕様(RR)またはサービス要求(SS)を経て部分サービスグラフ(PSG)に変換される。このPSGはペトリネットを用いてグラフ表現される。ここで、ペトリネットを使用した理由は、単に状態を表現するだけでなく、発火させてトークンを移動させることにより状態の推移を明確にできることである。内線相互接続サービス(正常終了時)の場合のPSGを図2に示す。

このようにして次々と作成される複数個の断片的なPSGから1つの統合サービスグラフ(TSG)に統合する。このTSGが求める最終的設計仕様であり、やはりペトリネットによりビジュアルに表現される。図3は内線相互サービスに対するTSGを示すが、これは図2以外に相手話中時や呼出中途中放棄を示すPSGも包含したものである。

このような統合は、1つのPSGをTSGとパターンマッチングさせ、TSGの中に存在しないものがあれば、これをTSGに付加することにより行われる。

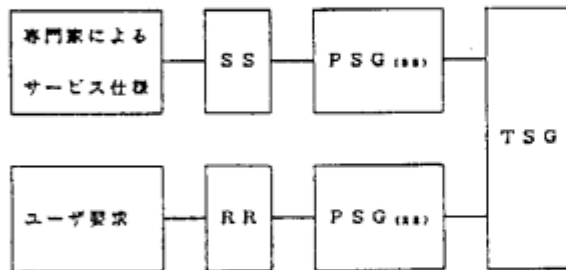
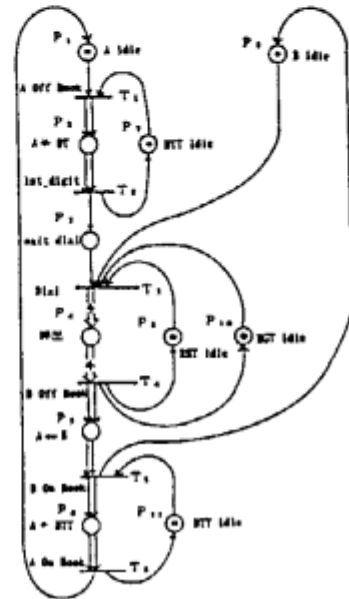


図1. 設計仕様の作成過程



3. サービスグラフの矛盾

全体として整合のとれたTSGを作成するには、それ自身に矛盾を含まないことが必要である。TSGに矛盾があるということは、2つの場合が考えられる。PSG自体が矛盾を有する場合と、PSG間に既に矛盾がありPSG-TSG変換の際に矛盾が作りこまれる場合である。PSG自体に矛盾があるとは以下の場合である。

- ①すべての成いは幾つかのトランジションを経由した後、初期状態(トークンの個数も含めて)に戻らない。(ループの関係)

なお、これは統合した結果のTSGに引き継がれ得るものであり、また新たに作成されることによって存在し得る矛盾である。

PSG間に矛盾があるとは以下の場合である。

- ②表現の詳しさがPSG間により異なる。(一方が詳細で、他方が省略等)

- ③入力プレースの集合と動作が一致し、出力プレースの集合が異なる。

上記の場合、それぞれPSGを作成する時、及び、TSGに統合する時に矛盾を除去することが必要である。

4. ベトリネットの解析による矛盾点除去

上記①及び③についてベトリネットの行列を用いて解析を行うことにより、設計仕様の検証・確認を行いつつPSG成いはTSGを作成してゆく。

ここで簡単にベトリネットの行列について述べる。⁽²⁾ トランジションの入力関数及び出力関数を表す行列 D^* 、 D^- を定義する。各行列はトランジションに対応する m 個の行とプレースに対応する n 個の列から成る。 D^- の j 行 i 列の要素は P_i から T_j に入るアークの本数を、また D^* の j 行 i 列の要素は T_j から P_i に出ていくアークの本数を示す。各プレースに存在するトークンの数を n 次元ベクトル μ (マーキングと呼ぶ)で表すと、(1)式が成立する時、 T_j が発火する。

$$\mu \geq e[j] \cdot D^- \quad \dots (1)$$

但し、 $e[j]$ は第 j 成分が1でそれ以外の成分が0の m 次元ベクトルである。

また、マーキング μ_0 において、 $T_{j_1}, T_{j_2}, \dots, T_{j_k}$ が発火すると、新しいマーキングは次のようになることが知られている。

$$\mu = \mu_0 + (e[j_1] + e[j_2] + \dots + e[j_k]) \cdot D \quad \dots (2)$$

但し、 $D = D^* - D^-$ 。

これを利用してまず①の検証を行うことができる。例えば図3の場合、行列 D は以下のように表される。

$$D = D^* - D^-$$

$$= \begin{bmatrix} -1 & 2 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & -1 & 4 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & -4 & 2 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & -2 & 2 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & 0 & 2 & 0 & 0 & 0 & -1 \\ 2 & 0 & 0 & -4 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

この時、初期状態から T_1, T_2, T_3, T_4 が発火すると元の状態に戻り、マーキングが変化しないことは下式により示される。

$$\begin{aligned} \mu &= \mu_0 + (1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1) \cdot D \\ &= \mu_0 + (0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0) \\ &= \mu_0 = (2 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1) \end{aligned}$$

前項でPSGからTSGを作成する手順について述べたが、同様にして入力関数行列及び出力関数行列についてもプレースやトランジションが一致しないものを追加してゆくことにより、最終的にはTSGの入力関数行列及び出力関数行列ができあがる。なお同種のリソースについては統合がなされる。このようにして作成されたTSGの2行列(D^* 及び D^-)をもとにループの確認を行える。つまり初期状態は各リソースがアイドル状態であるとし、そこから発火可能なトランジションを(1)式により探索し、それらを次々と発火させてゆくことにより検証が可能である。

また、そのようにして作成された断片的なサービスをCRT等に出力することにより、ユーザの気付かなかったサービスを提示することができる。そのサービスを許容するかどうかはユーザの判断に委ねる。

5. おわりに

ベトリネットを用いてビジュアルに表現した設計仕様に関数行列を使用した解析を行って矛盾の検出に適用した。総合的な設計仕様を作成する上で各種の矛盾を除去することは極めて重要であり、ここでとりあげた解析法は1つの手法である。行列を用いたベトリネットの解析には、行列は発火順序を規定しないこと、及び、そのすべての情報を表現していないこと等の問題点が残っており、検討の余地も大きい。

なお、本研究は第5世代コンピュータ・プロジェクトの一環として行っているものである。日頃ご指導を頂く岩下第5研究室長に感謝する。

6. 参考文献

- [1] 青柳、長谷川、田中、柴田：“通信システムにおける仕様設計エキスパートシステムの一検討”、信学研資(SE86-10)
- [2] J.L.Peterson、市川・小林訳：“ベトリネット入門”、共立出版(1984)
- [3] 田中、鹿野、山口：“通信システムに対するユーザ要求に含まれる矛盾の解消手法の一検討”、第33回情報処全大(5T-8)