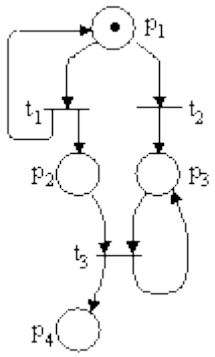
Sieci Petriego

Model podstawowy

Definicja sieci

Sieć Petriego jest czwórką $C = (P, T, I, O)$, w której $P = \{ p_1 , p_2 , ... , p_n \}$ jest skończonym zbiorem **miejsc**, $T = \{ t_1 , t_2 , ... , t_m \}$ jest skończonym zbiorem **tranzycji**, $I : T \rightarrow P^*$ jest funkcją wejściową, a $O : T \rightarrow P^*$ jest funkcją wyjściową. Zbiory miejsc i tranzycji są rozłączne $P \cap T = \Phi$.

Wartością funkcji $I (t_j)$ jest kolekcja miejsc wejściowych tranzycji t_j. Wyrażenie $\# (p_i , I (t_j))$ oznacza liczbę wystąpień miejsca p_i w kolekcji $I (t_j)$. Wartością funkcji $O (t_j)$ jest kolekcja miejsc wyjściowych tranzycji t_j. Wyrażenie $\# (p_i , O (t_j))$ oznacza liczbę wystąpień miejsca p_i w kolekcji $O (t_j)$.



Graficzną reprezentacją sieci jest graf dwudzielny, którego węzłami są miejsca, rysowane jako okręgi, i tranzycje, rysowane jako odcinki. Węzły grafu są połączone skierowanymi łukami w taki sposób, że żadne dwa miejsca i żadne dwie tranzycje nie są połączone bezpośrednio. Graf zawiera $\# (p_i , I (t_j))$ łuków od miejsca p_i do tranzycji t_j oraz $\# (p_i , O (t_j))$ łuków od tranzycji t_j do miejsca p_i.

Miejsca sieci mogą zawierać znaczniki, zaznaczane graficznie kropkami. Rozkład znaczników we wszystkich wyznacza znakowanie sieci, zdefiniowane jako funkcja: $\mu : P \rightarrow N$, ze zbioru miejsc w zbiór liczb naturalnych z zerem. Równoważnie, znakowanie można traktować jako wektor $\mu = (\mu (p_1), \mu (p_2), ... , \mu (p_n))$. W tym ujęciu zbiór wszystkich możliwych znakowań sieci jest równy N^n.

Znakowana sieć Petriego jest parą $Z = (C, \mu_0)$, w której C jest siecią Petriego, a $\mu_0 : P \rightarrow N$ jest znakowaniem początkowym.

Znakowanie sieci może ulec zmianie w wyniku odpalenia tranzycji. Warunkiem odpalenia jest wzbudzenie tranzycji, wyrażające się obecnością odpowiedniej liczby znaczników we wszystkich jej miejscach wejściowych. Odpalenie usuwa z każdego miejsca tyle znaczników, ile łuków prowadzi od tego miejsca do tranzycji, i deponuje w każdym miejscu tyle znaczników, ile łuków prowadzi od tranzycji do tego miejsca.

Tranzycja t_j jest **wzbudzona** (*enabled*) w znakowaniu μ, jeśli: $(\forall p_i \in P) [\# (p_i , I (t_j)) \leq \mu (p_i)]$

Odpalenie (*firing*) tranzycji t_j wzbudzonej w znakowaniu μ powoduje powstanie nowego znakowania μ' określonego przez warunek: $(\forall p_i \in P) [\mu' (p_i) = \mu (p_i) - \# (p_i , I (t_j)) + \# (p_i , O (t_j))]$

Odpalenie serii tranzycji, wzbudzonych w kolejno powstających znakowaniach poczynając od znakowania początkowego, jest nazywane wykonaniem znakowanej sieci (C, μ_0).

Wykonanie sieci produkuje ciąg kolejnych znakowań $(\mu_0 , \mu_1 , ... \mu_k , ...)$ oraz ciąg tranzycji $(t_{j0} , t_{j1} , ... t_{jk} , ...)$, taki że tranzycja t_{jk} jest wzbudzona w znakowaniu μ_k .

W ten sposób znakowana sieć Petriego tworzy maszynę abstrakcyjną o przestrzeni stanów N^n i funkcji przejścia $\delta : N^n \times T \rightarrow N^n$ określonej następująco:

1. Wartość funkcji $\delta (\mu , t_j)$ jest określona wtedy i tylko wtedy gdy $\# (p_i , I (t_j)) \leq \mu (p_i)$ — warunek wzbudzenia.
2. Jeśli wartość funkcji $\delta (\mu , t_j)$ jest określona, to $\delta (\mu , t_j) = \mu'$, gdzie: $\mu' (p_i) = \mu (p_i) - \# (p_i , I (t_j)) + \# (p_i , O (t_j))$ dla każdego $p_i \in P$ — definicja odpalenia.

Sekwencja znakowań $(\mu_0 , \mu_1 , ... \mu_k , ...)$ oraz sekwencja tranzycji $(t_{j0} , t_{j1} , ... t_{jk} , ...)$ powstające podczas wykonanie sieci spełniają warunek: $\mu_{k+1} = \delta (\mu_k , t_{jk})$.

Zbiór znakowań osiągalnych (*reachability set*) jest to najmniejszy zbiór $R (C, \mu_0)$, taki że:

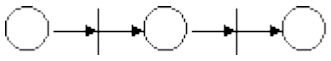
1. $\mu_0 \in R (C, \mu_0)$
2. Jeśli $\mu_l \in R (C, \mu_0)$ oraz istnieje tranzycja $t_j \in T$ taka, że $\mu_k = \delta (\mu_l , tj)$ to $\mu_k \in R (C, \mu_0)$

Zastosowania sieci

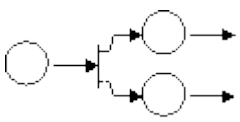
W informatyce sieci Petriego wykorzystuje się do modelowania synchronizacji i komunikacji procesów współbieżnych. W takim zastosowaniu miejsca sieci interpretuje się zazwyczaj jako warunki (lub stany) programu, a tranzycje jako akcje, np. instrukcje lub funkcje. Struktura sieci odwzorowuje wtedy strukturę programów, a ruch znaczników w sieci modeluje postępujące wykonanie procesów.

Zastosowanie sieci Petriego nie jest jednak ograniczone do tej jednej dziedziny. Abstrakcyjna definicja sieci jest w pełni ogólna i może być wykorzystana do modelowania procesów zachodzących w innych dziedzinach.

(a) Sieć Petriego jest uznanym modelem przepływu sterowania w systemach współbieżnych. Podstawowe konstrukcje sieci pozwalają łatwo modelować podstawowe konstrukcje sterujące:

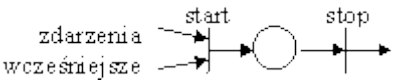

sekwencyjne wykonanie operacji,


alternatywę (z niedeterministycznym wyborem drogi),


rozszczepienie akcji na równoległe ścieżki wykonania.

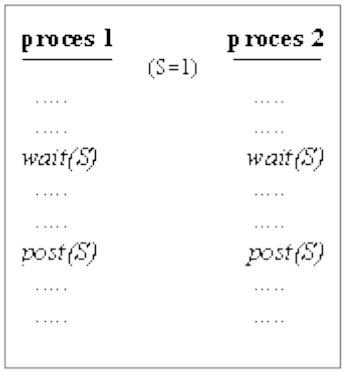
(b) Odpalenie tranzycji jest akcją atomową, tzn. modeluje zdarzenie proste. Sieć Petriego umożliwia jednak również modelowanie zdarzeń złożonych, charakteryzowanych przez

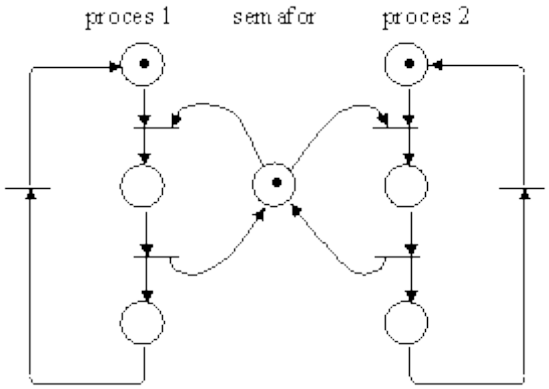
moment startu, moment zakończenia oraz okres trwania, podczas którego mogą zachodzić inne zdarzenia. Zajście zdarzenia może być uzależnione od zajścia zdarzeń wcześniejszych:



zdarzenie złożone

(c) Pojedynczy znacznik rezydujący w miejscu z kilkoma łukami wyjściowymi prowadzącymi do różnych tranzycji może wzbudzić wszystkie tranzycje wyjściowe, jednak odpalić będzie mogła tylko jedna z nich. Korzystając z tego można modelować konflikty procesów w dostępie do rzadkich zasobów systemu:





W podobny sposób można modelować inne uzależnienia wynikające z synchronizacji procesów, takie jak np. przekazywanie danych przez kolejki wiadomości lub komunikację przez spotkanie.

(d) Modelowanie złożonych problemów prowadzi do powstania bardzo dużych sieci, niemożliwych do przedstawienia graficznego. Opanowanie złożoności rysunku jest możliwe dzięki hierarchicznej reprezentacji sieci, w której całe podsieci widoczne na diagramach niższego poziomu są zastępowane pojedynczymi miejscami lub pojedynczymi tranzycjami na rysunkach wyższego poziomu, pokazujących problem na wyższym poziomie abstrakcji.

(e) Najpopularniejszymi modelami przepływu sterowania są automaty skończone i sieci działań. Semantyka tych modeli obejmuje jednak tylko systemy sekwencyjne. Semantyka sieci Petriego wychodzi poza systemy sekwencyjne i umożliwia modelowanie synchronizacji systemów równoległych. Dla każdego automatu lub sieci działań można skonstruować równoważny im model sieci Petriego.

(f) Sieci Petriego stosuje się również do modelowania innych systemów, niż systemy komputerowe. Przykładami mogą być uzależnienia proceduralne w systemach prawnych lub zależności między fazami procesu technologicznego, a dostępnością maszyn na liniach produkcyjnych.

Podsumowanie: zastosowania sieci Petriego w Informatyce obejmują:

- jednoznaczny opis semantyki (np. specyfikacji programu lub protokołu komunikacyjnego),
- makietowanie (symulacja wykonania sieci),
- analizę właściwości, weryfikacja (dowodzenie) poprawności programów.

Właściwości sieci

Zachowanie znakowanej sieci Petriego zależy od jej budowy i od znakowania początkowego. W różnych zastosowaniach sieci znaczenie mogą mieć różne aspekty jej działania. Tradycyjnie już wyróżnia się pewien zestaw właściwości, które można przekonująco interpretować w dziedzinach, z których wywodzą się modelowane systemy, a których rozstrzygniecie jest przedmiotem analizy sieci.

Bezpieczeństwo *(safeness)*

Miejsce $p_i \in P$ jest bezpieczne, jeśli: $(\forall \mu_k \in R (C, \mu_0)) [\mu_k (p_i) \leq 1]$
Sieć (C, μ_0) jest bezpieczna, jeśli wszystkie jej miejsca są bezpieczne.

W dowolnym znakowaniu osiągalnym miejsca sieci bezpiecznej mogą zawierać co najwyżej jeden znacznik. Miejsca sieci bezpiecznej modelują więc warunki boolowskie, które mogą być albo spełnione (znacznik obecny), albo nie (brak znacznika). Zachowanie sieci bezpiecznej może być łatwo modelowane w układach sprzętowych, w których każdemu miejscu odpowiada jeden przerzutnik.

Ograniczoność *(boundedness)*

Miejsce $p_i \in P$ jest q-ograniczone, jeśli $(\forall \mu_k \in R (C, \mu_0)) [\mu_k (p_i) \leq q]$
Sieć (C, μ_0) jest ograniczona, jeśli wszystkie jej miejsca są q-ograniczone dla pewnego q.

W dowolnym znakowaniu osiągalnym miejsca sieci ograniczonej mogą zawierać co najwyżej skończoną liczbę znaczników. Zbiór wszystkich możliwych znakowań sieci ograniczonej jest więc skończony, co gwarantuje pełną analizowalność sieci — każda właściwość może być rozstrzygnięta przez przeszukanie (skończonego) zbioru znakowań.

Zachowawczość *(conservation)*

Sieć (C, μ_0) jest ściśle zachowawcza, jeśli $(\forall \mu_k \in R (C, \mu_0)) [\Sigma_{(i \leq n)} \mu_k (p_i) = \Sigma_{(i \leq n)} \mu_0 (p_i)]$
Sieć (C, μ_0) jest zachowawcza, jeśli istnieje taki dodatni wektor $w = (w_1, \ldots , w_n)$, że:
$(\forall \mu_k \in R (C, \mu_0)) [\Sigma_{(i \leq n)} w_i \times \mu_k (p_i) = \Sigma_{(i \leq n)} w_i \times \mu_0 (p_i)]$

Sieć jest ściśle zachowawcza, jeśli liczba znaczników we wszystkich znakowaniach osiągalnych jest stała. Sięc jest zachowawcza, jeśli podczas wykonania znaczniki mogą rozszczepiać się i łączyć ponownie, powracając do tej samej liczby. Zachowawczość jest nieodzowną właściwością sieci modelującej konflikty zasobowe występujące podczas wykonania programów. Liczba znaczników w niektórych miejscach sieci modeluje tu liczbę dostępnych jednostek zasobowych systemu, która nie ulega zmianie w wyniku wykonania operacji przydzielenia lub zwolnienia zasobu.

Żywotność *(liveness)*

Tranzycja $t_j \in T$ jest żywa, jeśli $(\forall \mu_k \in R (C, \mu_0)) (\exists \mu \in R (C, \mu_k)) [(\mu , t) \in Dom (\delta)]$
Sieć (C, μ_0) jest żywa, jeśli każda jej tranzycja jest żywa.

Tranzycja jest żywa, jeśli z dowolnego znakowania osiągalnego sieci można dojść do takiego znakowania, w którym ta tranzycja będzie mogła odpalić. Sieć jest żywa, jeśli dla każdej tranzycji można z każdego znakowania osiągalnego dojść do takiego znakowania, w którym ta tranzycja będzie mogła odpalić. W odniesieniu do oprogramowania właściwość ta gwarantuje brak martwego kodu w każdym stanie wykonania programu.

Zakleszczenie *(deadlock)*

Sieć (C, μ_0) ma zakleszczenie, jeśli $(\exists \mu \in R (C, \mu_0)) (\forall t \in T) ((\mu , t) \notin Dom (\delta))$
Sieć jest wolna od zakleszczeń, jeśli: $(\forall \mu \in R (C, \mu_0)) (\exists t \in T) ((\mu , t) \in Dom (\delta))$

Zakleszczenie oznacza niemożliwość odpalenia jakiejkolwiek tranzycji. W odniesieniu do oprogramowania właściwość ta oznacza zawieszenie się wszystkich procesów programu.

Osiągalność znakowania *(reachability)*

(a) Problem osiągalności:
"Czy możliwe jest osiągnięcie wskazanego znakowania μ_k, tzn. czy $\mu_k \in R (C, \mu_0)$?"

(b) Problem pokrycia:
"Czy dla danego znakowania μ_k istnieje $\mu_l \in R (C, \mu_0)$ takie, że $\mu_l \geq \mu_k$?"

(c) Osiągalność znakowania częściowego:
"Czy dla danego znakowania μ_x określonego na $P_x \subset P$ istnieje $\mu_l \in R (C, \mu_0)$ takie, że $\mu_x = \mu_l \mid P_x$?"

Analiza sieci

Podstawową metodą analizy sieci Petriego jest budowa i analiza drzewa osiągalności. Inne metody, w tym przede wszystkim równania macierzowe i analiza niezmienników, mają mniejsze znaczenie.

Drzewo osiągalności *(reachability tree)*

Drzewo osiągalności jest grafem skierowanym, reprezentującym w skończony sposób wszystkie wykonania sieci (być może nieskończone). Węzłami drzewa są znakowania osiągalne, a łukami są odpalenia tranzycji. Korzeniem drzewa jest znakowanie początkowe, a jego liśćmi są znakowania końcowe lub znakowania powtórzone. Konstrukcja drzewa rozpoczyna się od znakowania początkowego i polega na obliczaniu kolejnych znakowań sieci, przez odpalanie wszystkich wzbudzonych w danym znakowaniu tranzycji. Ograniczenie rozmiarów drzewa zapewnia wprowadzenie specjalnego symbolu ω, oznaczającego możliwość gromadzenia się w miejscu sieci dowolnie dużej liczby znaczników. Jeśli w którejkolwiek gałęzi drzewa $\mu_0 \ldots \mu_k \ldots \mu_l \ldots$ wystąpi znakowanie większe od poprzedniego, tzn. takie w którym:
$(\forall p \in P) (\mu_l (p) \geq \mu_k (p)) \; \& \; (\exists p_i \in P) (\mu_l (p_i) > \mu_k (p_i))$
to przyjmuje się: $\mu_l (p_i) = \omega$. Właściwości symbolu ω:
$\omega + a = \omega$
$\omega - a = \omega$

$a < \omega$
$\omega \leq \omega$

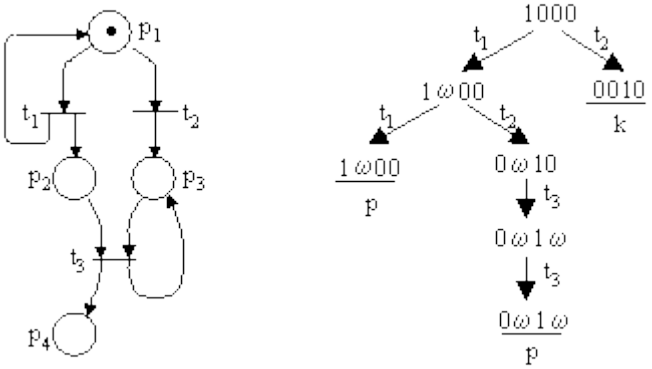
Algorytm konstrukcji drzewa

W opisie algorytmu przyjęta jest następująca klasyfikacja znakowań: znakowanie graniczne — nowe znakowanie oczekujące na analizę, znakowanie wewnętrzne — znakowanie już poddane analizie, znakowanie końcowe — znakowanie bez wzbudzonych tranzycji (liść drzewa), znakowanie powtórzone — znakowanie, które było już wcześniej analizowane w konstrukcji drzewa (liść).

1. Utwórz korzeń drzewa jako znakowanie graniczne μ_0
2. Dla każdego znakowania granicznego μ_x :

   - jeśli brak tranzycji wzbudzonych w znakowaniu μ_x, to μ_x jest znakowaniem końcowym (liść);
   - jeśli w drzewie istnieje już znakowanie μ_y , które nie jest znakowaniem granicznym i $\mu_y = \mu_x$, to μ_x jest znakowaniem powtórzonym (liść);
   - jeśli μ_x nie jest ani znakowaniem końcowym ani powtórzonym, to dla każdej wzbudzonej tranzycji t oblicz $\delta(\mu_x, t)$ i utwórz znakowanie μ_z w następujący sposób:
     — jeśli $\mu_x(p) = \omega$, to również $\mu_z(p) = \omega$,
     — jeśli w drodze od korzenia do μ_x istnieje znakowanie μ_y takie, że $\delta(\mu_x, t) \geq \mu_y$ oraz $\delta(\mu_x, t)(p) > \mu_y(p)$, to $\mu_z(p) = \omega$
     — w każdym innym przypadku $\mu_z(p) = \delta(\mu_x, t)(p)$

     Znakowanie μ_x staje się teraz znakowaniem wewnętrznym, a znakowanie μ_z granicznym.



Zastosowanie drzewa

Drzewo osiągalności jest skończoną reprezentacją nieskończonego zbioru osiągalności, nie może więc zawierać całej informacji o tym zbiorze. Dlatego nie wszystkie właściwości i problemy badawcze dotyczące sieci Petriego można rozstrzygnąć w oparciu o analizę drzewa.

<u>Bezpieczeństwo i ograniczoność</u>

Sieć jest ograniczona, jeśli w drzewie osiągalności nie występuje symbol ω. Wystąpienie symbolu ω wskazuje na możliwość nieograniczonej kumulacji znaczników, przy czym pozycja symbolu wskazuje miejsce, które jest nieograniczone.

Zachowawczość

Mając zadany wektor *w* można obliczyć sumy ważone wszystkich znakowań drzewa i sprawdzić, czy są one równe. Jeśli na którejś pozycji znakowania występuje symbol ω, to sieć może być zachowawcza tylko wtedy, gdy odpowiadająca mu składowa wektora *w* ma wartość *0* (ściśle biorąc, definicja zachowawczości wymaga $w_i > 0$ dla każdego *i*).

Nie znając wektora *w* można dla każdego węzła *j*, $j = 1 .. k$, drzewa osiągalności ułożyć równanie:
$\Sigma_{(i \le n)} w_i \times \mu_j(p_i) = s$

Otrzymując w rezultacie układ *k* równań liniowych i *n* nierówności z *n+1* niewiadomymi:

$\Sigma_{(i \le n)} w_i \times \mu_j(p_i) = s \qquad j = 1 .. k$
$w_i > 0 \qquad i = 1 .. n$

Sieć jest zachowawcza, jeśli ten układ ma rozwiązanie. Rozwiązaniami układu są współczynniki wektora *w* oraz wartość sumy ważonej *s*. Warunkiem zachowawczości sieci jest brak symbolu ω w drzewie osiągalności.

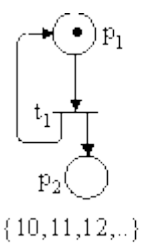
Problem pokrycia

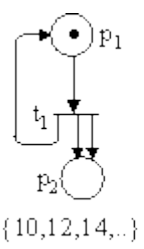
Problem pokrycia można rozstrzygnąć przez przegląd drzewa i sprawdzenie, czy dla zadanego znakowania μ_k istnieje w drzewie pokrywające je znakowanie . Jeśli w pokrywającym znakowaniu μ_l występuje kilka symboli ω to odrębnym problemem staje się znalezienie sekwencji odpaleń prowadzącej od znakowania początkowego do znakowania μ_l . (Np. jaka sekwencja odpaleń prowadzi do znakowania 0ω 1ω pokrywającego znakowanie 0612 ?)

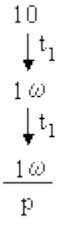
Ograniczenia drzewa

Symbol ω uniemożliwia rozróżnienie wszystkich stanów i rozstrzygnięcie problemów osiągalności i żywotności sieci. Istnieją tu jednak dwa ważne wyjątki:

1. Jeśli jakieś znakowanie nie jest pokryte, to nie jest też osiągalne.
2. Jeśli w drzewie osiągalności występuje węzeł końcowy, to sieć ma zakleszczenie.

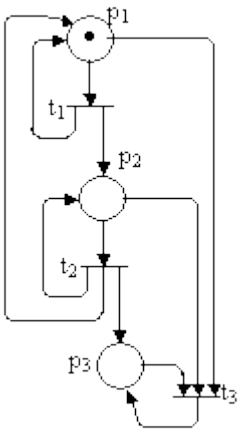


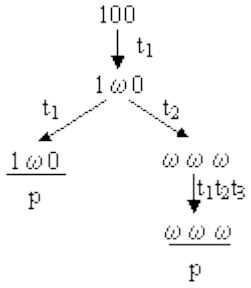




Przykłady

Sieci pokazane na rysunku mają takie samo drzewo osiągalności. Jednak zbiory osiągalności tych sieci (opisane poniżej grafów) są istotnie różne.





Drzewo osiągalności sieci pokazanej na rysunku nie zawiera węzła końcowego, jednak sekwencja odpaleń:
$100 \rightarrow 110 \rightarrow 211 \rightarrow 101 \rightarrow 111 \rightarrow 001$
prowadzi do zakleszczenia.

Zalety drzewa

Do zalet drzewa osiągalności należą: prosta koncepcja oraz możliwość automatycznej budowy i badania drzewa.

Twierdzenie

Drzewo osiągalności dowolnej sieci Petriego jest skończone.

Dowód:

Lemat 1: Każde nieskończone drzewo skierowane, w którym każdy węzeł ma skończoną liczbę bezpośrednich następników, zawiera nieskończoną gałąź zaczynającą się w korzeniu.

Dowód lematu 1 (przez konstrukcję). Zaczynamy od korzenia. Ponieważ ma on tylko skończoną liczbę następników, a całe drzewo jest nieskończone, więc przynajmniej jeden z następników musi być korzeniem poddrzewa nieskończonego. Rozważmy to poddrzewo. Tak jak poprzednio, przynajmniej jeden z następników korzenia tego poddrzewa musi być korzeniem poddrzewa nieskończonego. I tak dalej ... Przesuwając się od korzenia do korzeni kolejnych pod-drzew otrzymujemy gałąź nieskończoną.

Lemat 2: Każdy nieskończony ciąg nieujemny liczb całkowitych zawiera nieskończony podciąg niemalejący.

Dowód lematu 2 (przez konstrukcję). Możliwe są dwa przypadki:

1. Pewien element x_0 występuje w ciągu nieskończenie często. W takim przypadku podciąg $\{ x_0 , x_0 , ... \}$ jest szukanym nieskończonym podciągiem niemalejącym.
2. Brak elementów występujących nieskończenie często. Bierzemy dowolny element x_0. Istnieje co najwyżej x_0 elementów mniejszych i każdy występuje skończoną liczbę razy. Zatem ciąg zawiera jakiś element $x_1 \geq x_0$. Kontynuując, musi istnieć $x_2 \geq x_1$, itd.

Lemat 3: Każdy nieskończony ciąg n-wektorów nad zbiorem liczb całkowitych nieujemnych plus symbol ω zawiera nieskończony podciąg niemalejący.

Dowód lematu 3 przez indukcję ze względu na wymiar wektora.

1. ($n = 1$) Jeśli symbol ω występuje nieskończenie często, to podciąg $\{ \omega \}$ jest szukanym nieskończonym podciągiem niemalejącym. Jeśli nie, to usuwamy symbole ω i teza wynika z lematu 2.

2. ($n \Rightarrow n+1$) Rozważmy tylko pierwszy element ($n+1$)-wektorów i wybierzmy nieskończony podciąg wektorów z niemalejącym pierwszym elementem (jak w kroku 1). Zastosujmy do tego podciągu przesłankę indukcyjną, ignorując pierwszy element i wybierając z otrzymanego w ten sposób ciągu *n*-wektorów nieskończony podciąg niemalejący na wszystkich *n* elementach. Wybrany podciąg jest niemalejący zarówno na pierwszej składowej, jak i na pozostałych *n* składowych.

Dowód twierdzenia przez zaprzeczenie. Załóżmy, że drzewo osiągalności jest nieskończone. Na mocy lematu 1 drzewo zawiera nieskończoną gałąź wiodącą od korzenia $\{ \mu_0 , \mu_1, ... \}$. Ciąg znakowań $\{ \mu_0 , \mu_1 ... \}$ jest nieskończonym ciągiem *n*-wektorów. Na mocy lematu 3 istnieje w nim nieskończony podciąg niemalejący . Ale z konstrukcji drzewa $\mu_i \neq \mu_j$, bo inaczej μ_j byłoby znakowaniem powtórzonym kończącym gałąź. Zatem podciąg $\{ \mu_{i0} , \mu_{i1} ... \}$ jest ściśle rosnący: $\mu_{i0} < \mu_{i1} < ...$. Ale, znów z konstrukcji, jeśli $\mu_t < \mu_j$, to przynajmniej jedna ze skończonych składowych μ_j jest zastępowana symbolem ω. W rezultacie μ_{i1} zawiera co najmniej 1 symbol ω, μ_{i2} co najmniej 2 symbole ω, a μ_{in} zawiera same symbole ω. Ale wtedy $\mu_{in} = \mu_{i(n+1)}$ co oznacza, że $\mu_{i(n+1)}$ jest znakowaniem powtórzonym kończącym gałąź. Nieskończona gałąź nie istnieje.

Równania macierzowe

Sieć Petriego można reprezentować w postaci dwóch macierzy: D^-, D^+, których kolumny *1...n* odpowiadają miejscom, a wiersze *1...m* tranzycjom. Komórki macierzy D^- reprezentują funkcję wejściową *I*, a komórki macierzy D^+ reprezentują funkcję wyjściową *O*, tzn.:
$D^-(j, i) = \#(pi, I(tj))$
$D^+(j, i) = \#(pi, O(tj))$

Tranzycję reprezentuje jednostkowy *m*-wektor wierszowy $e(j)$. Znakowanie sieci jest *n*-wektorem wierszowym μ. Warunek wzbudzenia tranzycji t_j w znakowaniu μ ma postać:
$\mu \geq e(j) \times D^-$

Wynik odpalenia tranzycji t_j w znakowaniu μ opisuje funkcja przejścia δ:
$\delta(\mu, tj) = \mu - e(j) \times D^- + e(j) \times D^+ = \mu + e(j) \times (D^+ - D^-) = \mu + e(j) \times D$

gdzie $D = D^+ - D^-$. Wynik odpalenia sekwencji tranzycji $\sigma = (t_{j1}, t_{j2}, ..., t_{jk})$:
$\delta(\mu, \sigma) = \mu + e(j1) \cdot D + e(j2) \cdot D + ... + e(jk) \cdot D =$
$= \mu + (e(j1) + e(j2) + ... + e(jk)) \times D = \mu + f(\sigma) \times D$

gdzie: $f(\sigma)_j$ oznacza liczbę odpaleń tranzycj t_j (*Parikh mapping*).

Zastosowanie macierzy

Macierzowa reprezentacja sieci Petriego jest wygodna w obliczeniach komputerowych. Umożliwia łatwe badanie zachowawczości sieci oraz pozwala sformułować warunek konieczny (ale nie dostateczny) osiągalności znakowania.

Zachowawczość

Sieć jest zachowawcza jeśli: $(\exists w)(\forall \mu \in R(C, \mu_0))(\mu_0 \times w^T = \mu \times w^T)$. Skoro $\mu \in R(C, \mu_0)$, to istnieje sekwencja odpaleń σ taka, że: $\mu = \mu_0 + f(\sigma) \times D$. Podstawiając wartość μ

do poprzedniego równania otrzymujemy:
$\mu_0 \times w^T = \mu \times w^T = (\mu_0 + f (\sigma) \times D) \cdot w^T = \mu_0 \times w^T + f (\sigma) \times D \times w^T$

Odejmując stronami wyrażenie $\mu_0 \times w^T$ dochodzimy do równania $f (\sigma) \times D \times w^T = 0$,
które musi być spełnione dla każdego $f (\sigma)$

Ostatecznie otrzymujemy układ równań: $D \times w^T = 0$,
który ma rozwiązanie *w* wtedy i tylko wtedy, gdy sieć jest zachowawcza.

Osiągalność

Jeżeli $\mu \in R (C, \mu_0)$ to istnieje sekwencja odpaleń σ taka, że $f (\sigma) = x$ jest rozwiązaniem równania $\mu = \mu_0 + x \times D$. Zatem jeżeli znakowanie μ jest osiągalne, to równanie $\mu = \mu_0 + x \times D$ ma rozwiązanie.

Niezmienniki miejsc

Metoda niezmienników polega na określeniu zbioru formuł, opisujących pewne stałe (niezmienne) cechy sieci i osiąganych przez tę sieć znakowań, a następnie wnioskowaniu o właściwościach sieci na podstawie analizy tych formuł. Poszczególne formuły są na ogół prawdziwe w pewnym lokalnym fragmencie sieci, a ich prawdziwość wynika ze sposobu konstrukcji sieci, który z kolei odzwierciedla właściwości modelowanego za pomocą tej sieci systemu.

Sposób wykorzystania metody można prześledzić analizując przykład synchronizacji dostępu dwóch procesów do pojedynczego zasobu za pomocą semafora. Budując drzewo osiągalności sieci łatwo sprawdzić, że synchronizacja jest poprawna w takim sensie, że obydwa procesy nie mogą na raz wejść do swoich sekcji krytycznych. Za pomocą drzewa osiągalności nie można natomiast rozstrzygnąć, czy rozważana sieć ma zakleszczenie.

Niezmienniki

(1) $\mu (p_3) + \mu (p_4) + \mu (s) = 1$ — semafor
(2) $\mu (p_1) + \mu (p_3) = 1$ — proces 1
(3) $\mu (p_2) + \mu (p_4) = 1$ — proces 2

Analiza niezmienników

Przypuśćmy, że sieć ma zakleszczenie i rozważmy wszystkie możliwe rozkłady znaczników. Z (1) wynika, że dokładnie jeden znacznik musi rezydować w miejscu p_3 lub p_5 lub *s*. Jeśli ten znacznik jest w miejscu p_3 lub p_5 , to wzbudzona jest tranzycja t_3 lub t_4 , co prowadzi do sprzeczności z przypuszczeniem. Zatem znacznik musi przebywać w miejscu *s*. Jeśli jednak znacznika nie ma w miejscu p_3 , to z (2) wynika, że musi być w miejscu p_1 . W takim przypadku — znaczniki obecne w miejscach *s* i p_1 — wzbudzona jest tranzycja t_1 , co podobnie jak poprzednio prowadzi do sprzeczności z przypuszczeniem. Zakleszczenie w sieci nie może wystąpić.

Zalety metody

Do zalet metody niezmienników należy nieograniczony zakres jej zastosowania i brak wrażliwości na parametry modelu, takie jak np. liczba miejsc sieci. Wadą jest brak możliwości algorytmizacji metody, gdyż wymyślenie zbioru niezmienników wymaga twórczego wysiłku człowieka.

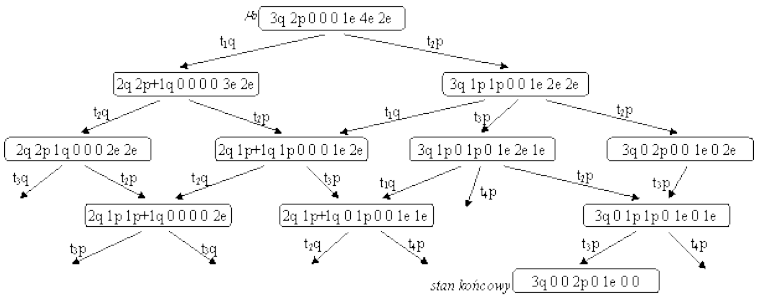
Kolorowane sieci Petriego

Wprowadzenie

Klasyczne sieci Petriego są modelem przepływu sterowania, pozbawionym możliwości opisu przetwarzania danych i wpływu wyników tego przetwarzania na przebieg procesu obliczeniowego. Kolorowane sieci Petriego rozszerzają zakres modelu wprowadzając do niego wartości, przypisane do krążących w sieci znaczników, oraz funkcje sterujące przepływem tych znaczników i obliczające nowe wartości.

Kolorowana sieć Petriego (*Coloured Petri Net*) jest grafem złożonym z **miejsc**, **tranzycji** i **łuków**, w którym poruszają się **znaczniki** przenoszące wartości określonego typu. W każdym miejscu sieci może znajdować się wiele różnych znaczników, których typ musi być jednak zgodny z typem tego miejsca. Wzbudzenie tranzycji w bieżącym znakowaniu sieci zależy zarówno od obecności znaczników w miejscach wejściowych tranzycji, jak i od wartości przypisanych do tych znaczników. Odpalenie tranzycji powoduje usunięcie pewnej liczby znaczników z miejsc wejściowych tranzycji i utworzenie pewnej liczby nowych znaczników w jej miejscach wyjściowych.

Zgodnie z tradycyjną terminologią kolorowanych sieci Petriego typ danych nazywa się **paletą kolorów**, a aktualna wartość przypisana do znacznika — **kolorem**. Znaczniki różnych kolorów, zgromadzone w pewnym miejscu sieci, tworzą kolekcję znaczników zapisywaną w postaci wyrażenia: $n_1'a_1 + n_2'a_2 + \ldots$, w którym symbole a_1 , a_2 oznaczają kolory, a symbole n_1 , n_2 liczby znaczników w poszczególnych kolorach.



Liczbę i kolor znaczników usuwanych w chwili odpalenia tranzycji definiują **wyrażenia wejściowe**, przypisane do jej łuków wejściowych. Liczbę i kolor znaczników tworzonych w wyniku odpalenia tranzycji definiują **wyrażenia wyjściowe**, przypisane do jej łuków wyjściowych. Dodatkowy warunek wzbudzenia określa przypisane do tranzycji wyrażenie, nazywane **dozorem**.

Formalna definicja modelu

Kolorowana sieć Petriego jest uporządkowaną dziewiątką $CPN = (\Sigma , P, T, A, N, C, G, E, \mu_0)$, w której:

Σ jest skończonym zbiorem palet kolorów (typów danych);
P jest skończonym zbiorem miejsc;
T jest skończonym zbiorem tranzycji, $P \cap T = \Phi$;
A jest skończonym zbiorem łuków, $A \cap P = \Phi$, $A \cap T = \Phi$;
N: $A \to (P \times T) \cup (T \times P)$ jest funkcją określającą miejsce łuków w sieci;
C: $P \to \Sigma$ jest funkcją przypisującą miejscom palety kolorów;
G: $T \to expressions$, jest funkcją przypisującą tranzycjom dozory;
E: $A \to expressions$ ($E (a) \in C (p (a))^*$) jest funkcja przypisującą łukom wyrażenia;
μ_0: $P \to expressions$ ($\mu_0 (p) \in C (p)^*$) jest znakowaniem początkowym.

Oznaczenia

$p (a)$ — miejsce początkowe lub końcowe łuku a .

$Var (t)$ — zbiór zmiennych tranzycji t, tzn. zbiór zmiennych związanych występujących w dozorze tranzycji oraz w wyrażeniach jej łuków wejściowych i wyjściowych; może się zdarzyć $Var (t) = \Phi$.

$Type (v)$ — typ zmiennej v , $Type (v) \in \Sigma$.

$E (p, t) = \sum_{(a: N (a) = (p, t))} E (a)$ — kolekcja znaczników usuwanych z miejsca p w chwili odpalenia tranzycji t .

$BT (t) = Type (v_1) \times Type (v_2) \times ... \times Type (v_n)$ — typ wiązania tranzycji t, tzn. zbiór wszystkich wiązań zmiennych tranzycji t (np. na poprzednim rysunku $BT (t_1) = U \times I$).

$B (t) = \{ (c_1 , c_2 , ... , c_n) \in BT (t): G (t) (c_1 , c_2 , ... , c_n) = True \} \subseteq BT (t)$ — zbiór wiązań dopuszczalnych tranzycji t, tzn. zbiór wiązań zmiennych spełniających dozór tranzycji (np. na poprzednim rysunku $B (t_1) = \{ (q,0) , (q,1) , (q,2) , ... \}$).

$B (T) = \sum_{(t \in T)} B (t)$ — przestrzeń wiązań sieci (suma zbiorów wiązań dopuszczalnych wszystkich tranzycji).

Znakowanie sieci

Znakowanie kolorowanej sieci Petriego jest funkcją $\mu : P \to \Sigma^*$, przypisującą każdemu miejscu kolekcję rezydujących w tym miejscu znaczników $(\forall p \in P) [\mu (p) \in C (p)^*]$

Odpalenie tranzycji

Każde wiązanie zmiennych $b \in B (t)$ wyznacza wartość wyrażeń przypisanych do łuków wejściowych i łuków wyjściowych tranzycji t. Tym samym wiązanie b wyznacza kolekcje znaczników, których obecność w miejscach wejściowych jest warunkiem wzbudzenia tranzycji t, oraz kolekcje znaczników, które zostaną zdeponowane w miejscach wyjściowych w wyniku odpalenia tej tranzycji.

Tranzycja t z wiązaniem zmiennych b jest wzbudzona w znakowaniu μ, jeśli:
$(\forall p \in P) [E (p, t) (b) \leq \mu (p)]$

Odpalenie tranzycji t z wiązaniem zmiennych b wzbudzonej w znakowaniu μ powoduje powstanie nowego znakowania μ' określonego przez warunek:
$(\forall p \in P)\ [\ \mu'(p) = \mu(p) - E(p,t)(b) + E(t,p)(b)\ ]$

Krok odpalenia

Krok odpalenia (*step*) jest funkcją częściową $Y : T \rightarrow B(T)^*$ przypisującą tranzycjom kolekcje dopuszczalnych wiązań zmiennych tych tranzycji $(\forall t \in T)\ (Y(t) \in B(t)^*)$.

Przypisana tranzycji t kolekcja wiązań zmiennych $Y(t)$ wyznacza wiele takich kolekcji znaczników, których obecność w miejscach wejściowych umożliwia wielokrotne odpalenie tej tranzycji. Krok Y określa kolekcje znaczników, które rozmieszczone w miejscach sieci umożliwiają wielokrotne odpalenie tranzycji należących do dziedziny funkcji Y.

Przyjmując oznaczenie: $b \in Y(t) \Leftrightarrow (t, b) \in Y$ można zapisać warunek wzbudzenie kroku Y w znakowaniu μ:
$(\forall p \in P)\ [\ \sum_{((t,b) \in Y)} E(p,t)(b) \leq \mu(p)\ ]$

Wynikiem odpalenia kroku Y wzbudzonego w znakowaniu μ jest nowe znakowanie μ' określone jako:
$\mu'(p) = \mu(p) - \sum_{((t,b) \in Y)} E(p,t)(b) + \sum_{((t,b) \in Y)} E(t,p)(b)$

Wykonanie sekwencji kroków $Y_1, Y_2, \ldots, Y_k \ldots$ prowadzi do powstania sekwencji znakowań osiągalnych:
$\mu_0\ [Y_1 > \mu_1\ [Y_2 > \mu_2 \ldots \mu_{k-1}\ [Y_k > \mu_k \ldots$

Na przykład, jeśli dla sieci z poprzedniego rysunku określimy krok $Y = \{ (t_1, (q,0)), (t_1, (q,1)), (t_1, (q,1)) \}$, to krok ten będzie wzbudzony w każdym znakowaniu μ, w którym:
$\mu(A) \geq ((q,0) + 2'(q,1))$
$\mu(R) \geq 3'e$
$\mu(S) \geq 3'e$

Wynikiem odpalenia kroku Y będzie usunięcie tych znaczników z miejsc A, S, T i zdeponowanie kolekcji znaczników $(q,0) + 2'(q,1)$ w miejscu B.

Analiza i weryfikacja modelu

Każdą sieć kolorowaną można przekształcić w równoważną jej sieć Petriego, którą z kolei można analizować za pomocą standardowych metod analizy sieci. Stwierdzenie to, ważne z teoretycznego punktu widzenia, ma jednak niewielką przydatność praktyczną, gdyż sieć Petriego równoważna sieci kolorowanej jest na ogół olbrzymia. (Podobne znaczenie ma twierdzenie, że każdy problem z zakresu arytmetyki liczb całkowitych można wyrazić w języku maszyny Turinga.)

Realne możliwości formalnej analizy sieci kolorowanych są, niestety, ograniczone. Rozszerzenia wprowadzone do pierwotnego modelu sieci Petrigo zwiększają wprawdzie siłę wyrazu tego modelu, ale ograniczają możliwości jego analizy. Nie udało się skonstruować ani odpowiednika drzewa osiągalności sieci kolorowanej, ani reprezentacji macierzowej.

Jedynymi narzędziami analizy pozostała metoda niezmienników oraz analiza grafu osiągalności, który jednak w ogólnym przypadku może być nieograniczony.

Niezmienniki sieci kolorowanej buduje się zliczając (ważąc) znaczniki różnych kolorów. Dla przykładu przedstawimy dowód braku zakleszczenia kolorowanej sieci Petriego przedstawionej na poprzednim rysunku.

Niezmienniki

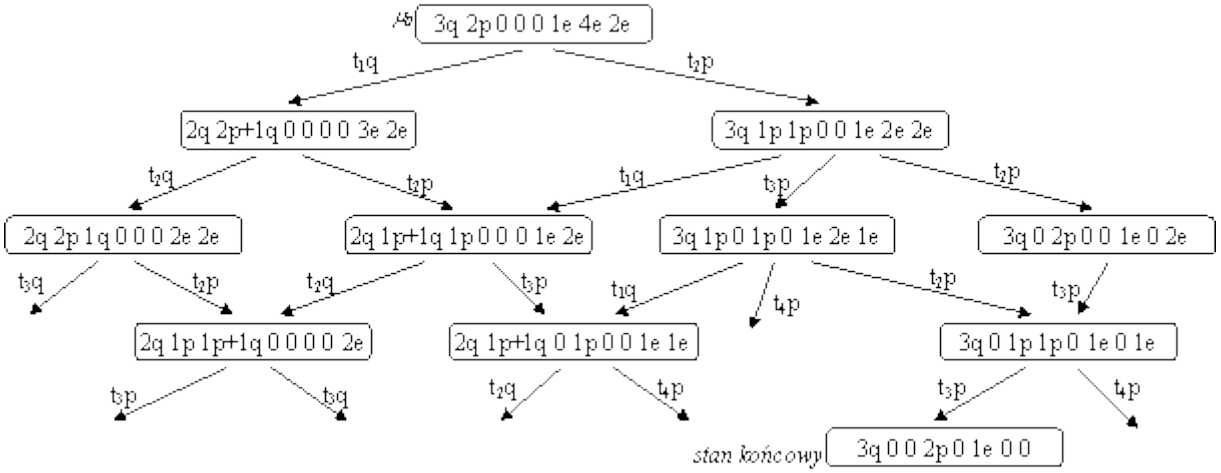
(1) $P(\mu(B) + \mu(C) + \mu(D) + \mu(E)) = 2$ — procesy typu p
(2) $Q(\mu(A) + \mu(B) + \mu(C) + \mu(D) + \mu(E) = 3$ — procesy typu q
(3) $E(\mu(R)) + Q(\mu(B) + \mu(C)) = 1$ — zasób R
(4) $E(\mu(S)) + Q(\mu(B)) + 2 \cdot PQ(\mu(C) + \mu(D) + \mu(E)) = 3$ — zasób S
(5) $E(\mu(T)) + P(\mu(D)) + PQ(\mu(E)) + P(\mu(E)) = 2$ — zasób T

Analiza niezmienników

Przypuśćmy, że sieć ma zakleszczenie i rozważmy wszystkie możliwe rozkłady znaczników. Z (1) wynika, że dokładnie 2 *p*-znaczniki muszą przebywać w miejscach *B* – *E*. Z (2) wynika, że dokładnie 3 *q*-znaczniki muszą przebywać w miejscach *A* – *E*. Jeśli jednak jakiś znacznik znajduje się w miejscu *E*, to wzbudzona jest tranzycja t_5 , co prowadzi do sprzeczności z przypuszczeniem. Z kolei z (4) wynika, że w miejscach *C* i *D* może przebywać łącznie co najwyżej jeden znacznik. W takim razie z (5) otrzymujemy, że przynajmniej 1 znacznik musi znajdować się w miejscu *T* i wzbudzona jest tranzycja t_3 lub t_4 , co również prowadzi do sprzeczności z przypuszczeniem. Zatem wszystkie *p*-znaczniki muszą przebywać w miejscu *B*, a *q*-znaczniki w miejscach *A* i *B*. Jednak z (3) wynika, że w miejscu *B* może znajdować się co najwyżej 1 *q*-znacznik. W takim razie z (4) otrzymujemy, że co najmniej 2 znaczniki muszą przebywać miejscu *S*. To jednak oznacza, że wzbudzona jest tranzycja t_2 , co prowadzi do sprzeczności z przypuszczeniem. W ten sposób wyczerpane zostały wszystkie możliwe rozkłady znaczników, mogące prowadzić do zakleszczenia. Zakleszczenie sieci jest niemożliwe.

Analiza grafu osiągalności

Dodajmy do systemu modelowanego na poprzednim rysunku jedną jednostkę zasobu S (łącznie system ma teraz cztery jednostki tego zasobu) i ponownie rozważmy problem zakleszczenia sieci. Znakowanie początkowe można przedstawić jako wektor (dla uproszczenia pominięte są drugie składowe *p*-znaczników i *q*-znaczników):
$\mu_0 (A, B, C, D, E, R, S, T) = (3'q, 2'p, 0, 0, 0, 1'e, 4'e, 2'e)$.



Jak widać z rysunku, graf osiągalności sieci zawiera węzeł końcowy, co dowodzi, że sieć ma zakleszczenie. Zauważmy, że wynik ten nie jest zgodny z intuicją — zakleszczenie jest na ogół rezultatem braku zasobów, a nie ich nadmiaru, tymczasem tu zostało spowodowane dodaniem do systemu dodatkowej jednostki jednego z zasobów.

Hierarchizacja sieci

Kolorowane sieci Petriego mają reprezentację hierarchiczną, analogiczną do hierarchicznych diagramów DFD. Całe podsieci widoczne na diagramach niższego poziomu mogą być zastępowane pojedynczymi miejscami lub pojedynczymi tranzycjami na rysunkach wyższego poziomu, pokazujących problem na wyższym poziomie abstrakcji. Możliwe jest też zdefiniowanie wzorca podsieci i podstawianie różnych wystąpień tego wzorca w miejsce różnych miejsc lub tranzycji.

Hierarchizacja nie wnosi do modelu żadnej nowej semantyki — każdą sieć hierarchiczną można rozwinąć w równoważną jej sieć płaską.

Wykorzystanie sieci

Praktyczne wykorzystanie modeli formalnych wymaga posiadania odpowiednich systemów wspomagających projektowanie. Działająca w USA firma Meta Software Corporation oferuje komercyjny system CASE, w którego skład wchodzą trzy pakiety programowe:

1. Design/IDEF — wspomaganie analizy projektowania strukturalnego metodą SADT.
2. Design/CPN — automatyczna translacja diagramów SADT na kolorowaną sieć Petriego (transformacje → tranzycje, przepływy → miejsca i łuki, etykiety przepływów → palety kolorów, specyfikacje transformacji → wyrażenia).
3. Design/ML — automatyczna generacja kodu w języku funkcjonalnym ML.

Kolorowana sieć Petriego może być poddana ręcznej analizie i weryfikacji poprawności metodą niezmienników. Opublikowane przykłady zastosowania systemu dotyczą realistycznych problemów elektronicznego transferu pieniędzy i systemu radarowego nadzoru przestrzeni powietrznej. Wymagania dotyczące systemu transferu pieniędzy wymagały wykonania 10 000 – 30 000 transakcji/15 min. Wydajność osiągnięta na stacji SUN przez automatycznie wygenerowaną aplikację wyniosła ok. 20,000 transakcji/15min.