김일성종합대학학보 주체108(2019)년
정보과학 제65권 제2호

의존선택연산에 의한 π-계산론리의 문장론적확장과 그것을 리용한 업무통합체계의 모형화

박지혜, 김용석

선행연구[2]에서는 업무통합체계를 π-계산론리식으로 모형화하고 모형검증도구를 리용하여 설계의 정확성을 검증하였다. 그러나 체계의 분산성과 복잡성이 증가되는데 따라 π-계산론리에서 선택연산(+)은 론리식의 해석에서 계산량을 지수함수적으로 증가시켰다.

론문에서는 π-계산론리의 선택연산에 의하여 론리식해석에서 복잡성을 해소하기 위한 방법으로 처리들사이의 의존성에 기초한 의존선택연산을 새로 제안하여 π-계산론리의 문장론을 기호적으로 확장하였다. 또한 확장된 π-계산론리로 어느 한 건설기업소의 계획보고체계를 모형화함으로써 의존선택연산의 효과성을 분석하였다.

1. 의존선택연산

π-계산론리에서 선택연산(+)은 두 처리들사이의 비결정성선택실행을 의미한다. 즉 처리 P, Q에 대하여 $P+Q$는 처리 P가 선택되든가 또는 처리 Q가 선택된다는것을 의미한다. 이러한 비결정성선택으로 하여 모의기나 검증기에서 두가지 실행경로를 다 론의하여야 한다. 그러므로 π-계산론리식에 선택연산의 수가 많을수록 계산량이 지수함수적으로 증가하게 된다.

실례로 다음과 같은 론리식에서 가능한 실행경로를 론의하자.

$$(P+Q)\cdot(R+S)$$

가능한 실행경로는 $P\cdot R$, $P\cdot S$, $Q\cdot R$, $Q\cdot S$이다. 이 4개의 경우에 대하여 모의 혹은 검증을 진행하여야 한다. 그러나 처리 $P+Q$와 $R+S$가 의존성을 가진다면 실행경로가 2개로 될수 있다. 즉 $P+Q$에서 처리 P가 선택될 때 $R+S$에서는 처리 R가 선택되여야 한다고 보면 가능한 실행경로는 $P\cdot R$, $Q\cdot S$이다.

업무체계개발에서 이러한 요구사항은 많이 제기된다. 그러므로 π-계산론리식의 작성에서 처리들사이의 의존성관계를 잘 리용하면 계산량을 지수함수적으로 줄일수 있다.

의존선택연산은 처리들사이의 의존성을 나타내는 π-계산론리의 연산자이다.

체계의 복잡성을 줄이기 위해 보충선택(Complement Choice)과 배타적선택(Conjunctive Choice)을 정의하며 이것은 내부와 외부선택연산들의 의존성에 기초한다.

① 보충선택은 2개의 처리들 즉 하나의 처리가 다른 처리의 선택에 의존하면서 그것에 대한 보충적이며 결정적인 선택을 하도록 하는 선택의존성에 기초한다. 이것을 외부의존성이라고 한다.

② 배타적선택은 하나의 처리안에서의 선택의존성에 기초한다. 하나의 처리에는 수많은 선택연산들이 있을수 있으며 그 선택연산들가운데 일부 의존성이 있을수 있다. 이것을

내부의존성이라고 한다.

배타적선택연산은 선택연산에 의하여 초래되는 처리의 복잡도를 줄이며 따라서 체계의 복잡도도 지수함수적으로 줄일수 있다. 또한 보충선택연산은 체계의 복잡도를 줄인다.

· 보충선택

2개의 처리 P와 Q를 가지고있는 체계를 다음과 같이 가정한다.

$$P ::= (A + \overline{A}).P$$

$$Q ::= (B + \overline{B}).Q$$

선택연산의 의존성에 의하여 처리 P가 A를 선택한다고 할 때 Q가 반드시 B를 선택하여야 한다고 가정하면 보충선택연산은 다음과 같이 표시된다.

$$P ::= (A \oplus^j \overline{A}).P$$

$$Q ::= (B \oplus^j \overline{B}).Q$$

보충선택연산자를 리용한 경우와 리용하지 않은 경우에 체계의 상태해석을 비교하면 다음과 같다.

$$P \times Q ::= (A + \overline{A}) \times (B + \overline{B}) = A \times B + A \times \overline{B} + \overline{A} \times B + \overline{A} \times \overline{B}$$

$$P \times Q ::= (A \oplus^j \overline{A}) \times (B \oplus^j \overline{B}) = A \times B + \overline{A} \times \overline{B}$$

이처럼 선택연산은 체계의 상태를 표현하는데서 복잡성을 지수함수적으로 증가시킨다. 그리고 보충선택연산을 리용하여 복잡도를 줄일수 있다.

· 배타적선택

처리 P에 대하여 배타적선택연산은 다음과 같이 표시된다.

$$P = (A_1 \oplus_i A_2) \cdots (B_1 \oplus_i B_2)$$

배타적선택연산은 $\oplus_i$로 표시하며 여기서 i는 내부의존성의 구분을 위해 리용되는 첨수이다. 배타적선택연산에서는 식의 순서에 따라 의존성이 적용된다. 만약 A_1을 선택하였다면 그것은 연산자의 왼쪽 항이므로 뒤에 있는 같은 첨수의 배타적선택연산에서도 연산자의 왼쪽 항인 B_1이 선택된다.

일반적으로 선택연산은 매 처리들에 대하여 독립적인 특성을 가지며 앞의 선택연산이 뒤의 선택연산에 영향을 주지 않을수도 있다. 그러나 일부 요구명세서에서는 선택연산들사이의 의존성을 필요로 하는 경우가 있다.

처리 P가 $(A_1 + A_2)$에서 A_1을 선택하였을 때 처리 Q도 $(B_1 + B_2)$에서 B_1을 선택해야 한다고 가정하자. 기존의 처리대수에서는 이러한 처리들사이의 의존성을 표현하는 서술능력이 없다. 그러므로 이것을 해결하기 위하여 선택연산전에 체계의 선택에 따라 처리를 진행할수 있도록 그것을 알려주는 어떤 통보문을 주고받는 등의 부분을 추가하여야 한다. 이것은 복잡도를 증가시키는 원인으로 된다. 즉 배타적선택연산을 리용하지 않았을 때에는 체계의 복잡도가 지수함수적으로 늘어날수 있으며 의존성이 필요한 부분에서는 교착상태(Deadlock)가 발생할수 있다.

의존성은 연산자의 첨수에 의하여 구분되므로 의존성이 복잡하게 얽혀있더라도 첨수를 통하여 단순하게 표현할수 있다. 그리고 매 선택에 따르는 경우의 수에 의하여 식을 분할할 필요가 없다.

보충선택연산과 배타적선택연산을 통털어 의존선택연산이라고 하며 이 연산은 다음

과 같은 성질을 가진다.

· $A \oplus_i B \neq B \oplus_i A$

· $(A \oplus_i B) \times (C \oplus_i D) = (A \times C) + (B \times D)$

· $(A \oplus_i B) \times (C \oplus_j D) = (A \times (C \oplus_j D)) + (B \times (C \oplus_j D))$

2．의존선택연산의 응용실례

론문에서는 π－계산론리의 의존선택연산[1]을 계획보고체계(Plan Report System)[3]에 적용하였다.

어느 한 건설기업소는 사업소 1, 사업소 2, 사업소 3 등 17개의 부분사업소들로 구성되였다. 매 사업소들은 건설계획에 따라 시공계획과 자재타산계획을 세워 계획과장에게 제출한다. 계획과는 계획과장외에 5명의 부원으로 구성되였다.

계획과에 제출된 건설계획은 과의 부원들에 의하여 검토되고 해당 계획들이 시공과와 자재과에 보내진다.

론문에서는 사업소 1과 2, 2명의 계획부원과 계획과장에 대해서만 론의한다.

결국 체계는 다음과 같이 구성된다.

① 계획작성자 4명(시공계획작성자(PC) 2명, 자재계획작성자(PM) 2명)

② 계획부원 2명(부원 1(MEM1), 부원 2(MEM2))

③ 계획과장(계획과(DEP))

④ 건설사업소 2개(사업소 1($L1$), 사업소 2($L2$))

⑤ 시공과(DC)와 자재과(DM)

매 사업소에는 시공계획작성자와 자재계획작성자가 각각 1명씩 있고 2명의 계획부원들은 계획과장의 지시를 받는다.

체계의 요구명세서는 다음과 같다.

① 시공계획은 시공과에, 자재계획은 자재과에 가져가야 한다.

② 매 계획부원은 최대로 2개의 계획을 받고 해당 부문에 가져갈수 있다.

체계의 일반적인 업무흐름은 계획작성자가 먼저 계획과에 계획을 제출하겠다는 요청을 보내고 계획과장은 대기상태에 있는 계획부원에게 지령을 주며 계획부원은 계획작성자가 있는 사업소에 가서 계획을 받아가지고 해당한 과(시공과 혹은 자재과)에로 가져간다. 체계를 보다 단순하게 하기 위하여 계획작성자와 그가 작성한 계획을 하나의 대상으로 본다. 여기서 계획부원이 계획을 검토하는 처리는 내부처리로 보고 체계서술에서는 론의하지 않는다.

이때 요청의 경우는 다음과 같이 결정된다.

① 두 사업소에서 같은 계획작성자들이 동시에 요청하는 경우 계획부원은 두 계획을 다 받아가지고 해당한 과에로 가져간다.

② 한 사업소에서 서로 다른 계획작성자들이 동시에 요청하는 경우 계획부원은 그 사업소에 가서 두가지 계획들을 받아가지고 각각 해당한 과에로 가져간다.

③ 한 사업소에서 한 계획작성자만이 요청하는 경우 계획부원은 이 계획만을 받아가지고 해당한 과에로 가져간다.

이로부터 체계는 확장된 π-계산론리식으로 다음과 같이 모형화한다.

$$\text{in}\,P\cdot\text{get}\,Q\cdot\text{out}\,P=\text{io}\,P(Q)$$

$$\text{in}\,P\cdot\text{put}\,Q\cdot\text{out}\,P=\text{io}\,P(\overline{Q})$$

$$\text{out}\,P\cdot\text{get}\,Q\cdot\text{in}\,P=\text{oi}\,P(Q)$$

$$\text{out}\,P\cdot\text{put}\,Q\cdot\text{in}\,P=\text{oi}\,P(\overline{Q})$$

$$\text{PC}=L1[P(\langle\overline{C},\overline{L1},\overline{\text{PC}},\overline{\text{PM}}\rangle)\cdot(\text{MEM1get}\cdot\text{MEM1put}+\text{MEM2get}\cdot\text{MEM2put})]$$

$$\text{PM}=L1[(\text{MEM1get}\cdot\text{MEM1put}+\text{MEM2get}\cdot\text{MEM2put})]$$

$$\text{PC}=L2[P(\langle\overline{C},\overline{L2},\overline{\text{PC}},\overline{\text{PM}}\rangle)\cdot(\text{MEM1get}\cdot\text{MEM1put}+\text{MEM2get}\cdot\text{MEM2put})]$$

$$\text{DEP}=P(\langle C,L1,\text{PC},\text{PM}\rangle)\cdot A(\langle\overline{R},\overline{L1}\rangle)\cdot P(\langle C,L2,\text{PC},\text{PM}\rangle)\cdot A(\langle\overline{R},\overline{L2}\rangle)\cdot$$
$$\cdot(A(\overline{\text{get}\,CM})\oplus_2^1(A(\overline{\text{get}\,C})\oplus_2^2A(\overline{\text{get}\,M})))\cdot(A(\overline{\text{get}\,CM})\oplus_1(A(\overline{\text{get}\,M})\oplus_2A(\overline{\text{get}\,C})))$$

$$\text{MEM1}=A(R)\cdot\text{out DEP}\cdot(A(L1)\oplus_3^3A(L2))\cdot(A(\text{get}\,CM)\oplus_1^1(A(\text{get}\,C)\oplus_2^2A(\text{get}\,M)))\cdot$$
$$\cdot((\text{io}\,L1(\text{PC},\text{PM})\oplus_3\text{io}\,L2(\text{PC},\text{PM}))\oplus_1((\text{io}\,L1(\text{PC})\oplus_3$$
$$\text{io}\,L2(\text{PC}))\oplus_2(\text{io}\,L1(\text{PM})\oplus_3\text{io}\,L2(\text{PM})))\cdot$$
$$\cdot((\text{io DC}(\overline{\text{PC}})\oplus_3\text{io DM}(\overline{\text{PM}}))\oplus_1((\text{io}\,L2(PC)\oplus_3$$
$$\text{io}\,L1(\text{PC}))\oplus_2(\text{io}\,L2(\text{PM})\oplus_3\text{io}\,L1(\text{PM})))\cdot$$
$$\cdot((\text{io DM}(\overline{\text{PM}})\oplus_3\text{io DC}(\overline{\text{PC}}))\oplus_1(\text{io DC}(\overline{\text{PC}},\overline{\text{PC}})\oplus_2$$
$$\text{io DM}(\overline{\text{PM}},\overline{\text{PM}})))\cdot\text{in DEP}$$

$$\text{MEM2}=A(R)\cdot\text{out DEP}\cdot(A(L2)\oplus_3^3A(L1))\cdot(A(\text{get CM})\oplus_1^1(A(\text{get M})\oplus_2^2A(\text{get}\,C)))\cdot$$
$$\cdot((\text{io}\,L2(\text{PC},\text{PM})\oplus_3\text{io}\,L1(\text{PC},\text{PM}))\oplus_1((\text{io}\,L2(\text{PM})\oplus_3$$
$$\text{io}\,L1(\text{PM}))\oplus_2(\text{io}\,L2(\text{PC})\oplus_3\text{io}\,L1(\text{PC})))\cdot$$
$$\cdot((\text{io DM}(\overline{\text{PM}})\oplus_3\text{io DC}(\overline{PC}))\oplus_1((\text{io}\,L1(\text{PM})\oplus_3$$
$$\text{io}\,L2(\text{PM}))\oplus_2(\text{io}\,L1(\text{PC})\oplus_3\text{io}\,L2(\text{PC})))\cdot$$
$$\cdot((\text{io DC}(\overline{\text{PC}})\oplus_3\text{io DM}(\overline{\text{PM}}))\oplus_1(\text{io DM}(\overline{\text{PM}},\overline{\text{PM}})\oplus_2$$
$$\text{io DC}(\overline{\text{PC}},\overline{\text{PC}})))\cdot\text{in DEP}$$

여기서 get는 계획을 받는 행위, put는 계획을 주는 행위, in P는 P지역으로 들어가는 행위, out P는 P지역에서 벗어나는 행위이며 일부 표시들을 추가하였다.

모형화된 체계의 π-계산론리식에서 볼수 있는것처럼 의존선택연산이 DEP에서는 2개, MEM1과 MEM2에서는 3개의 류형이 리용되였다.

이제 DEP와 MEM1, MEM2에서 의존선택연산과 일반선택연산을 리용하는 경우 실행경로의 수를 계산하여 비교하자.

① DEP

일반선택연산을 리용한 경우 실행경로의 수는

$$(1+(1+1))\times(1+(1+1))=9$$

이다. 이 경우

$$(1\oplus_1(1\oplus_2 1))\times(1\oplus_1(1\oplus_2 1))=(1\times1)+((1\oplus_2 1)\times(1\oplus_2 1))=$$
$$=1+((1\times1)+(1\times1))=1+(1+1)=3$$

이다.

DEP의 추상그라프를 다음의 그림에 보여주었다.

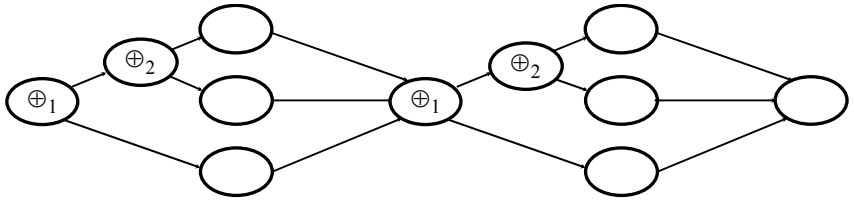

그림. DEP의 추상그라프

그림에서 보여준것처럼 의존선택연산을 리용하면 9개의 경우가 3개의 경우로 감소된다는것을 알수 있다.

② MEM1

일반선택연산을 리용한 경우

$$(1+1)\times(1+(1+1))\times((1+1)+((1+1)+(1+1)))\times((1+1)+((1+1)+(1+1)))\times$$
$$\times((1+1)+(1+1))=2\times3\times6\times6\times4=864$$

이다.

의존선택연산을 리용한 경우 우와 마찬가지로 계산하면 6이다.

일반선택연산을 리용한 경우 체계에서 실행경로의 수는 $9\times864\times864=6\ 718\ 464$ 이다.

의존선택연산을 리용한 경우 $3\times6\times6=108$ 이다.

3．효과성분석

의존선택연산에 의하여 체계를 모형화하는데서 실행경로의 수가 얼마나 줄어들었는가를 다음의 표에 보여주었다.

표. 일반선택연산과 의존선택연산에서 실행경로의 수

부분품	일반선택연산	의존선택연산
DEP	9	3
MEM1	864	6
MEM2	864	6
SYSTEM	6 718 464	108

표에서 보여준것처럼 의존선택연산에 의하여 체계의 실행경로의 수가 지수함수적으로 감소한다는것을 알수 있다.

한편 모의할 때 선택연산을 정확히 리용하지 못하면 교착상태가 발생할수 있다. 이 교착상태는 계획과장이 보내려고 하는 지령과 계획부원이 받으려고 하는 지령이 일치하지 않는 경우, 계획부원이 계획을 가져가려고 할 때 해당 계획이 이미 해당한 과에 보내진 경우 등으로 하여 동기화가 되지 않는 상태를 의미한다. 이러한 교착상태가 발생하면 즉시 체계는 정지하게 되며 그뒤에 있게 될 모든 선택연산이 실행되지 않고 종료된다. 따라서 수많은 경우의 수가 실행되지 않고 종료된다. 그러므로 계산을 통해 얻은 결과인 6 718 464보다 훨씬 적은 수의 경로만이 발생하게 된다. 이러한 실행경로의 수도 적지 않으며 체계의 해석을 위해서는 경우의 수를 좀더 줄일 필요성이 있다.

의존선택연산을 리용하여 일부 실행경로들을 제한하기때문에 모든 선택에 있어서 체계가 정상적으로 실행될수 있는 경우만을 선택하게 된다. 그러므로 실행의 결과 교착상태가 발생하지 않으므로 정상적인 실행이 불가능한 모든 선택에 대한 경로가 실행되지 않으며 정상적인 실행결과만이 나타나게 된다.

맺 는 말

π－계산론리의 선택연산에 의한 론리식해석에서의 복잡성을 해소하기 위한 방법으로서 처리들사이의 의존성에 기초한 의존선택연산을 새로 제안하여 π－계산론리의 문장론을 기호적으로 확장하였다. 확장된 π－계산론리로 어느 한 기업소의 계획보고체계를 모형화함으로써 의존선택연산의 효과성을 분석하였다.

참 고 문 헌

[1] **김일성**종합대학학보(자연과학), 63 , 10 , 32, 주체106(2017).

[2] 김성련, 리준식; 론리와 응용, **김일성**종합대학출판사, 80～150, 주체95(2006).

[3] L. Jianxiao et al.; Using Pi-calculus to Model Web Service Interaction, 9, 5, 1759, 2013.

주체108(2019)년 2월 5일 원고접수

An Extension of Formal Syntax of π-calculus with Conjunctive and Complement Choice and its use in Modelling Business Combination System

Pak Ji Hye, *Kim Yong Sok*

In this paper, we introduce new notion of conjunctive and complement choice which reduces system complexity and propose a method to model business combination system.

Key words: conjunctive and complement choice, business combination system, π-calculus