튜·토·리·얼

다익스트라의 '프로그래밍의 수련(修練)': 일곱 번째, 적절히 종료(終了)하는 구조(構造)의 설계에 대하여 (Dijkstra's "A Discipline of Programming": The Seventh Lecture, On the Design of Properly Terminating Constructs)

김도형
성신여자대학교 컴퓨터정보학부
Do-Hyung Kim
School of CSE, Sungshin Women's University
E-mail: dkim@cs.sungshin.ac.kr; URL: http://cs.sungshin.ac.kr/~dkim

요약

이번 튜토리얼에서는 앞장에서 논의한 '반복 구조를 위한 기본 정리'의 결론이 되는 함축의 전제 중 일부인 wp(DO, T)에 관해 보다 깊게 논의한다. 비록 이것이 의미하는 바가 직관적으로는 단순히 '반복 구조가 종료할 최약 사전 조건'으로 간단하지만, 임의의 DO 구조가 주어졌을 때 이러한 조건을 구하는 것은 일반적으로 매우 어렵다. 따라서 반복 구조를 구성할 때 아예 처음부터 종료에 대한 조건을 염두에 두고 설계하는 것이 현실적인 방법이 된다. 기본적인 착상은 단순하다. 각각의 반복 구조에 대해 반복되는 동안 음이 아닌 값을 유지하면서 동시에 매 반복마다 1 이상 값이 감소되는 정수 함수를 하나씩 연계시키는 것이다. 이러한 함수가 관련 정리들의 전제를 모두 만족한다면, 거기에 대응되는 반복 구조는 적절히 종료하게 된다.

1)변하지 않고 참을 유지하는 조건 P에 대해, 반복 구조를 위한 기본 정리는

$$(P \text{ and } \mathrm{wp}(\mathrm{DO}, T)) \Rightarrow \mathrm{wp}(\mathrm{DO}, P \text{ and non } BB)$$

라고 주장한다.

여기에서 wp(DO, T) 항(項)은 반복 구조가 종료될 최약(最弱) 사전 조건이다. DO 구조가 아무렇게나 주어졌을 때, wp(DO, T)를 결정하는 일은 일반적으로—불가능은 아닐지 몰라도—매우 어렵다; 그러므로 나는 종료라는 요구조건을 (아예 처음부터) 염두에 두고 우리의 반복 구조를 설계하자고 제안한다. 즉, 종료를 위한 적절한 증명을 선택한 뒤 프로그램이 이 증명의 가정을 만족하도록 만들자는 것이다.

다시 P가 불변(invariant) 조건이라고 하자. 즉, 모든 상태에 대해

$$(P \text{ and } BB) \Rightarrow \mathrm{wp}(\mathrm{IF}, P) \quad (1)$$

라고 하고, 추가로 t가 모든 상태에 대해

$$(P \text{ and } BB) \Rightarrow (t > 0) \quad (2)$$

이 성립하는 유한(有限) 정수 함수라고 하고, 여기에 모든 상태에서 임의의 값 $t0$에 대해

$$(P \text{ and } BB \text{ and } t \leq t0+1) \Rightarrow \mathrm{wp}(\mathrm{IF}, t \leq t0) \quad (3)$$

1) 이번 강의는 다익스트라의 원저에서 제6장에 해당하는 부분이다.

이라고 하자. 그렇다면 우리는 모든 상태에 대해

$$P \Rightarrow \text{wp}(\text{DO}, T) \tag{4}$$

임을 증명할 것이며, 반복을 위한 기본 정리와 더불어 이것으로부터 우리는 모든 상태에 대해

$$P \Rightarrow \text{wp}(\text{DO}, P \text{ and non } BB) \tag{5}$$

라는 결론을 내릴 수 있다.[2)]

우리는 이것을 수학적 귀납법을 통해 모든 상태에서

$$(P \text{ and } t \leq k) \Rightarrow H_k(T) \tag{6}$$

가 $k \geq 0$인 모든 k에 대해 성립함을 증명함으로써 보인다. 우리는 먼저 $k = 0$에 대해 (6)의 참을 확인한다. $H_0(T) = \text{non } BB$이므로, 우리는 모든 상태에 대해

$$(P \text{ and } t \leq 0) \Rightarrow \text{non } BB \tag{7}$$

임을 증명하여야 한다. 그런데 (7)은 다름 아닌 (2)와 같은 식이다; 두 식 모두

$$\text{non } P \text{ or non } BB \text{ or } (t > 0)$$

과 같고,[3)] 따라서 (6)은 $k = 0$에 대해 성립한다.

이제 $k = K$에 대해 (6)이 성립한다고 가정한다; 그러면

$$\begin{aligned}(P \text{ and } BB \text{ and } t \leq K+1) &\Rightarrow \text{wp}(\text{IF}, P \text{ and } t \leq K) \\ &\Rightarrow \text{wp}(\text{IF}, H_K(T));\end{aligned} \quad 4)$$

$$\begin{aligned}(P \text{ and non } BB \text{ and } t \leq K+1) &\Rightarrow \text{non } BB \\ &= H_0(T)\end{aligned} \quad 5)$$

가 성립한다. 그리고 이 두 개의 함축(含蓄; implication)들은 다음:

$$\begin{aligned}(P \text{ and } t \leq K+1) \Rightarrow \text{wp}(\text{IF}, H_K(T)) \text{ or} \\ H_0(T) = H_{K+1}(T)\end{aligned}$$

과 같이 결합될 수 있고($A \Rightarrow C$와 $B \Rightarrow D$로부터 우리는 (A or B) $\Rightarrow$ (C or D)가 성립한다고 결론지을 수 있다), 따라서 (6)의 참은 $k \geq 0$인 모든 k에 대해 성립함이 확인되었다. t는 유한 함수이므로,

$$(\text{E } k: k \geq 0: t \leq k) \quad 6)$$

와

$$\begin{aligned}P &\Rightarrow (\text{E } k: k \geq 0: P \text{ and } t \leq k) \\ &\Rightarrow (\text{E } k: k \geq 0: H_k(T)) \quad 7) \\ &= \text{wp}(\text{DO}, T)\end{aligned}$$

---

2) 지난번 강의에서 다룬 '반복을 위한 기본 정리'는 모든 상태에서 (1) 식이 성립한다면

$$(P \text{ and } \text{wp}(\text{DO}, T)) \Rightarrow \text{wp}(\text{DO}, P \text{ and non } BB)$$

가 성립한다는 것이다. 따라서 (4) 식이 성립한다면 위의 함축에서 전제는 P만으로 줄어든다.

3) 잘 알다시피 함축 ($P \Rightarrow Q$)는 (**non** P **or** Q)와 동치이다. 또한 (**non** (P **and** Q))는 (**non** P **or non** Q)와 동치이다.

4) 이 식에서 첫 번째 함축은 (1) 식과 (3) 식, 그리고 술어 변환자의 성질 3에 의해 성립한다. 즉, (1) 식과 $t0$ 자리에 K를 넣은 (3) 식을 결합하면

$$\begin{aligned}&(P \text{ and } BB \text{ and } t \leq K + 1) \\ &\quad \Rightarrow \text{wp}(\text{IF}, P) \text{ and } \text{wp}(\text{IF}, t \leq K)\end{aligned}$$

가 성립하며, 여기에 술어 변환자의 세 번째 성질을 적용하면 된다. 두 번째 함축은 (6) 식의 귀납가정(induction hypothesis)과 술어 변환자의 성질 2에 의해 성립한다. 참고로 술어 변환자의 성질은 다익스트라의 원저 제3장에서 다루어졌다.

5) 이 식의 첫 번째 함축은 자명하다는 것을 노파심에 환기시킨다. ((P **and** Q) $\Rightarrow$ P)는 항상 성립하기 때문이다.

6) t가 유한 함수이므로 이 식은 당연히 성립한다.

7) 이 식의 첫 번째 함축은 t가 유한 함수라는 것으로부터 성립한다. t가 유한 함수이므로 t의 값보다 작지 않은 정수는 항상 존재한다. 즉, (**E** k: $k \geq 0$: $t \leq k$) = T이다. 따라서 ($P \Rightarrow$ (P **and** (**E** k: $k \geq 0$: $t \leq k$)))가 성립한다. 두 번째 함축은 (6) 식에

가 성립하며, 따라서 (4)가 증명되었다.

직관적으로 이 정리는 매우 명백하다. 한편으로는 P가 참으로 유지될 것이므로 $t \geq 0$도 역시 참으로 유지되며[8]; 다른 한편으로는 관계 (3)은 가드 명령을 매번 선택할 때마다 t가 적어도 1은 실질적으로 감소하게 만들 것이라는 점을 기술하고 있다. 가드 명령이 선택되는 횟수가 무제한적이라면 t를 제한 없이 감소시킬 것이며, 이것은 모순(矛盾)으로 이끌 것이다[9].

이 정리의 적용가능 여부는 (2)와 (3)의 성립 여부에 달려 있다. 관계 (2)는 다소 뻔한 것이나,[10] 관계 (3)은 약간 까다로운 면이 있다. 선택 구조를 위한 우리의 기본 정리에서

$$Q = (P \text{ and } BB \text{ and } t \leq t0+1)$$
$$R = (t \leq t0)$$

으로 하면—두 술어 모두에 자유 변수 $t0$이 나타난다는 것이 우리가 '술어 쌍'에 대해 언급해 온 이유이다—(그 정리는) 만약 다음

$$(\mathbf{A}\ j: 1 \leq j \leq n: (P \text{ and } B_j \text{ and } t \leq t0+1) \Rightarrow \text{wp}(SL_j, t \leq t0))$$

이 성립한다면 (3)도 성립한다는 결론을 우리가 내릴 수 있다고 말해 준다.[11] 달리 말하자면, 우리는 각 가드 명령에 대해서 그것을 선택하면 t의 실질적인 감소를 일으킬 것이라는 점을 증명해야 한다. 우리는 t가 현재 상태의 함수라는 것을 염두에 두고 다음 (술어)

$$\text{wp}(SL_j, t \leq t0) \tag{8}$$

을 고찰해 볼 수 있다. 이것은 상태 공간의 좌표 변수들 이외에도 자유 변수 $t0$ 또한 포함하고 있는 술어이다. 여태까지 우리는 이러한 술어를 상태들의 부분집합을 규정짓는 것으로서 간주해 왔다. 그러나 어떤 상태가 하나 주어졌을 때, 우리는 이 술어를 $t0$에 부여된 조건으로 취급할 수도 있다.[12] $t0 = tmin$을 식 (8)의 $t0$을 위한 최소의 해라고 하자; 그렇게 되면 우리는 값 $tmin$을 t의 최종값을 위한 최저상한(最低上限; lowest upper bound)으로서 해석할 수 있다. t 자신과 꼭 마찬가지로 $tmin$ 역시 현재 상태의 함수라는 점을 상기하면, 술어

$$tmin \leq t-1$$

은 SL_j가 실행되면 t의 값이 적어도 1은 감소된다는 것을 보장하는 최약 사전 조건으로 해석될 수

의한 것이며, 마지막 동치는 DO 구조의 의미 정의에 따른 것이다(다익스트라의 원저 제4장에 IF와 DO 구조의 의미 정의가 나온다).

8) 식 (1)과 (2)에 의해서 그러하다.

9) 식 (2)를 만족시키지 못 하기 때문이다. 어떤 정리가 자기 자신의 전제에 위배되므로 자가당착(自家撞着)이고 모순이다. 그 결과로 도출되는 결론은 DO 구조의 반복 횟수가 무제한적이 되지 않고, 언젠가는 '적절히 종료한다'는 것이다.

10) 관계 (2)를 만족하는 유한 정수 함수 t를 찾는 것은 전혀 어렵지 않다. 예컨대 $t = k$ $(k > 0)$와 같은 상수(常數) 함수는 관계 (2)를 만족시킨다.

11) 선택 구조를 위한 기본 정리에 따르면, 술어 쌍 Q와 R이 모든 상태에 대해

$$Q \Rightarrow BB$$

와

$$(\mathbf{A}\ j: 1 \leq j \leq n: (Q \text{ and } B_j) \Rightarrow \text{wp}(SL_j, R))$$

을 만족하면,

$$Q \Rightarrow \text{wp}(\text{IF}, R)$$

역시 모든 상태에 대해 성립한다는 것이다. 여기서 Q와 R을 본문에서처럼 정의하면 선택 구조를 위한 기본 정리의 첫 번째 조건

$$(P \text{ and } BB \text{ and } t \leq t0 + 1) \Rightarrow BB$$

는 당연히 만족되므로,

$$(\mathbf{A}\ j: 1 \leq j \leq n: (P \text{ and } B_j \text{ and } t \leq t0 + 1) \Rightarrow \text{wp}(SL_j, t \leq t0))$$

이 만족되면

$$(P \text{ and } BB \text{ and } t \leq t0 + 1) \Rightarrow \text{wp}(\text{IF}, t \leq t0),$$

즉 (3) 식이 만족된다.

12) 지금까지처럼 술어를 변수 t에 부여된 조건이 아니라, 이제 t가 하나의 상태에서 값이 고정된 경우이므로 $t0$에 부여된 조건으로 보자는 뜻이다.

있다. 여기서—반복하자면—두 번째 인자 t는 정수 값을 가지는 현재 상태의 함수인 이 사전 조건을

$$\mathrm{wdec}(SL_j, t)$$

로 나타내면, P의 불변과 t의 실질적인 감소는 모든 j에 대해

$$(P \textbf{ and } B_j) \Rightarrow (\mathrm{wp}(SL_j, P) \textbf{ and } \mathrm{wdec}(SL_j, t)) \quad (9)$$

가 성립하면 보장된다.

적당한 B_j를 찾기 위한 대개의 실제 방법은 다음과 같다. 식 (9)는

$$(P \textbf{ and } Q) \Rightarrow R$$

과 같은 형태이고, 여기서 주어진 P와 R에 대한 Q—실제 계산가능한(computable)!—를 하나 찾아야만 하는 것이다. 우리는 다음과

1. $Q = R$은 하나의 해(解)다.
2. 만약 $Q = (Q1 \textbf{ and } Q2)$가 하나의 해고 $P \Rightarrow Q2$라면, $Q1$ 역시 해다.
3. 만약 $Q = (Q1 \textbf{ or } Q2)$가 하나의 해고 $P \Rightarrow \textbf{non } Q2$라면(또는 결국 같은 것이 되지만 $(P \textbf{ and } Q2) = F$), $Q1$ 역시 해다.
4. 만약 Q가 하나의 해고 $Q1 \Rightarrow Q$라면, $Q1$ 역시 해다.

같은 점들을 살펴본다.

주의 1. 이렇게 하는 과정에서[13] 만약 우리가 $P \Rightarrow \textbf{non } Q$와 같은 B_j를 위한 Q의 후보에 이르렀다면, 이 후보 Q는 $Q = F$(앞의 단계 (3)에 따르면, 모든 Q에 대해 $Q = (F \textbf{ or } Q)$이므로)로 보다 더 간단해질 수 있다; 이것은 현재 고려중인 가드 명령이 쓸모가 없으며, 결코 선택되지 않을 것이기 때문에 (가드 명령) 집합에서 빼버릴 수 있음을 의미한다. (주의 1의 끝.)

13) 바로 앞 부분에서 이야기하고 있는 (9) 식을 만족하는 가드들 B_j를 구하는 과정을 뜻한다.

주의 2. 종종 (9) 식을 두 개의 식

$$(P \textbf{ and } B_j) \Rightarrow \mathrm{wp}(SL_j, P) \quad (9a)$$

와

$$(P \textbf{ and } B_j) \Rightarrow \mathrm{wdec}(SL_j, t) \quad (9b)$$

로 분리하여 별도로 다루는 것이 실용적이다. 따라서 두 개의 고려사항이 분리된 셈이다: (9a)는 불변으로 유지되는 것과 관련이 있고, 반면에 (9b)는 진행(進行)을 보장하는 것과 관련이 있다. 만약 식 (9a)를 다루는 도중에 $P \Rightarrow B_j$와 같은 B_j에 이르렀다면, 그러한 B_j를 가지고는 P의 불변성이 비종결(nontermination)을 보장할 것이기 때문에 이 조건[14]은 (9b)를 만족하지 않을 것이라는 점이 확실하다.[15] (주의 2의 끝.)

그러므로 우리는

$$P \Rightarrow \mathrm{wp}(\mathrm{DO}, P \textbf{ and non } BB)$$

와 같은 DO 메커니즘을 만들 수 있다. 우리가 선택한 B_j들은 함축 (9)를 만족시킬 수 있도록 충분히 강해야만 하며,[16] 그 결과로 이제 보장되는 사후 조건 P **and non** BB는 너무 약해져서 요구하는 사후 조건 R을 함축하지 못 할 지도 모른다.[17]

14) B_j를 뜻한다.

15) 만약 어떤 가드 B_j가 불변 조건 P가 성립하는 동안 항상 참을 유지한다면, 즉 $(P \Rightarrow B_j)$가 참이라면, 이러한 불변 조건과 가드를 가지는 반복 구조는 결코 끝나지 않는다. 참인 가드가 항상 하나 이상 존재하기 때문이다. 이런 경우 그러한 불변 조건과 가드는 당연히 (9b) 식을 만족하지 않을 것이다.

16) 함축이 성립하기 위해서는 전제를 이루는 조건이 강하면 강할수록 용이하다. 가장 강한—어떻게 해도 만족시킬 수 없을 정도로 강한—조건인 F가 전제이면, 결론을 이루는 조건에 관계없이 함축은 참이 되는 것이다. (9) 식의 전제를 이루는 B_j가 강하면 강할수록 그 함축은 참이 되기가 쉽다.

17) 반복 구조가 종료할 때, 우리는 불변 조건 P에 더해서 참인 가드가 하나도 없다는 조건, 즉 **non** BB를 부가적으로 만족하게 된다. 이 조건, 곧 $(P$ **and non** $BB)$가 우리가 원하는 사후 조건 R을 함축하면 우리가 원하는 바가 달성되는 것이다. 그런데

그런 경우, 우리는 아직 문제를 해결하지 못한 것이며 다른 가능성들을 고려해야 한다.

참고 문헌

[1] Bergin, T. J. and R. G. Gibson (Eds.), *History of Programming Languages*, Addison Wesley, New York, 1996.

[2] Dahl, O.-J. E. W. Dijkstra, and C. A. R. Hoare, *Structured Programming*, Academic Press, New York, 1972.

[3] Dijkstra, E. W., "Guarded Commands, Nondeterminacy, and Formal Derivation of Programs," *Communications of the ACM*, Vol. 18, No. 8, pp. 453-457, 1975.

[4] Dijkstra, E. W., *A Discipline of Programming*, Prentice Hall, Englewood Cliffs, NJ, 1976.

[5] Ghezzi, C. and M. Jazayeri, *Programming Language Concepts*, 2nd Ed., Wiley, New York, 1987.

[6] Hoare, C. A. R., "An Axiomatic Basis of Computer Programming," *Communications of the ACM*, Vol. 12, No. 10, pp. 576-580, 1969.

[7] Hoare, C. A. R., and N. Wirth, "An Axiomatic Definition of the Programming Language Pascal," *Acta Informatica*, Vol. 2, pp. 335-355, 1973.

[8] Hoare, C. A. R., "The Emperor's Old Clothes," *Communications of the ACM*, Vol. 24, No. 2, pp. 75-83, 1981.

[9] Kafura, D., *Object-Oriented Software Design and Construction with C++*, Prentice Hall, Upper Saddle River, NJ, 1998.

[10] Louden, K. C., *Programming Languages: Principles and Practice*, PWS Publishing Company, 1993.

[11] Sammet, J. E., "Programming Languages: History and Future," *Communications of the ACM*, Vol. 15, No. 7, pp. 601-610, 1972.

[12] Sebesta, R. W., *Concepts of Programming Languages*, 4th Ed., Addison Wesley Longman, 1999.

[13] Wexelblat, R. L. (Ed.), *History of Programming Languages*, Academic Press, New York, 1981.

$$BB = (\mathbf{E}\ j\colon 1 \le j \le n\colon B_j)$$
$$= (B_1 \textbf{ or } B_2 \textbf{ or } \cdots \textbf{ or } B_n)$$

이므로,

$$\textbf{non } BB = (\textbf{non } B_1 \textbf{ and non } B_2 \textbf{ and } \cdots \textbf{ and non } B_n)$$

이 되고, 가드 B_j가 강해지면 강해질수록 **non** BB는 약해진다. 따라서 함축

$$(P \textbf{ and non } BB) \Rightarrow R$$

이 성립되기 더 어려워지는 것이다.

필자 약력

김도형

1981년 ~ 1985년:
서울대학교 공과대학
컴퓨터공학과(학사)
1985년 ~ 1987년:
한국과학기술원
전산학과(석사)
1987년 ~ 1992년:
한국과학기술원
전산학과(박사)
1992년:
한국과학기술원 정보전자연구소 연수연구원
1992년 ~ 현재:
성신여자대학교 컴퓨터정보학부 부교수
1997년 ~ 1998년:
스토니 브룩 소재 뉴욕주립대학교
컴퓨터과학과 객원교수

관심분야:
프로그래밍 언어