# 다익스트라의 '프로그래밍의 수련(修練)': 여섯 번째, 두 개의 정리(定理)
# (Dijkstra's "A Discipline of Programming": The Sixth Lecture, Two Theorems)

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 두 개의 정리를 유도하고 증명한다. 이 정리들은 각각 대안적 구조와 반복적 구조를 위한 기본 정리라고 부르며 가드 명령 집합으로부터 구성한 문장들에 관한 것들이다. 간단하게 요약하자면, 대안적 구조를 위한 기본 정리는 선택 구조가 실행될 때 불변 조건에 관한 것이며, 반복적 구조를 위한 기본 정리는 대안적 구조를 위한 기본 정리가 성립한다는 전제 하에서 루프 구조가 실행을 종료할 때 달성되는 사후 조건에 관한 것이다.

이 장[1)]에서 우리는 가드 명령 집합으로부터 구성한 문장들과 관련하여 두 개의 정리를 도출(導出)한다. 보조적(補助的)인 정리는 대안(代案)을 나타내는(alternative) **if**-**fi**-구조를 다루고, 주(主)된 정리는 반복을 나타내는(repetitive) **do**-**od**-구조와 관계가 있다. 우리는 이 장에서 가드 명령 집합

$$B_1 \rightarrow SL_1 \mid B_2 \rightarrow SL_2 \mid \ldots \mid B_n \rightarrow SL_n$$

으로부터 유도되는 구조들을 논의할 것이다. 위의 가드 명령 집합을 '**if** . . . **fi**' 쌍이나 '**do** . . . **od**' 쌍으로 둘러싸서 구성되는 문장들을 각각 'IF'와 'DO'로 표시하겠다. 또한 다음

$$BB = (\mathbf{E}\, j\colon 1 \le j \le n\colon B_j)$$

과 같은 축약(縮約) 표현을 사용할 것이다.

1) 다익스트라의 책에 따르자면 제5장('Chapter 5. Two Theorems')이다.

**정리**

**대안적 구조를 위한 기본 정리**.

우리는 조금 전에 서술한 표기 약속을 사용하여 대안적 구조를 위한 기본 정리를 만들 수 있다:

대안적 구조 IF와 술어(述語; predicate)들의 어떤 쌍 $Q$와 $R$[2)]이 다음처럼 모든 상태에서

$$Q \Rightarrow BB \qquad (1)$$

와

$$(\mathbf{A}\, j\colon 1 \le j \le n\colon (Q \text{ and } B_j) \Rightarrow \mathrm{wp}(SL_j, R)) \qquad (2)$$

이 성립한다면,

2) '어떤 술어들 $Q$와 $R$'이 아니라, '술어들의 어떤 쌍 $Q$와 $R$'이라는 표현에 주의하기 바란다. 임의의 두 술어가 아니라, 어떤 상호 관계가 있는 두 술어들이다. 이 점은 이 장의 중반부와 다음 장에서 보다 분명해질 것이다.

$$Q \Rightarrow \text{wp}(\text{IF}, R) \qquad (3)$$

도 역시 모든 상태에서 성립한다.[3)]

정의에 의해

$$\text{wp}(\text{IF}, R) = BB \textbf{ and } (\mathbf{A}\, j: 1 \le j \le n: B_j \Rightarrow \text{wp}(SL_j, R))$$

이고 (1) 덕분에 우변의 첫 번째 항은 함축(含蓄; implication)되므로, 우리가 (2)를 사용하여

$$Q \Rightarrow (\mathbf{A}\, j: 1 \le j \le n: B_j \Rightarrow \text{wp}(SL_j, R)) \qquad (4)$$

이 모든 상태에서 성립한다고 결론 내릴 수 있다면 (3)은 증명된다. $Q$가 거짓인 임의의 상태에서 (4)는 함축의 정의에 의해 참이다.[4)] $Q$가 참인 상태에서 임의의 $j$에 대해 우리는 두 가지 경우를 나눈다: $B_j$는 거짓일 수 있다. 그런데 그렇게 되면 함축의 정의에 의해 $B_j \Rightarrow \text{wp}(SL_j, R)$은 참이 된다.[5)] 혹은 $B_j$는 참일 수 있다. 그런데 그 때는 (2) 때문에 $\text{wp}(SL_j, R)$은 참이 되고, 따라서 $B_j \Rightarrow \text{wp}(SL_j, R)$도 역시 참이 된다. 결과적으로 (4) 그리고 나아가서 (3)이 증명된다.

**주의.** 양자택일 ($n=2$)이고 $B_2 = \textbf{non } B_1$인 특별한 경우에는 $BB = T$이고 최약 사전 조건은

$$(B_1 \Rightarrow \text{wp}(SL_1, R)) \textbf{ and } (\textbf{non } B_1 \Rightarrow \text{wp}(SL_2, R)) =$$
$$(\textbf{non } B_1 \textbf{ or } \text{wp}(SL_1, R)) \textbf{ and } (B_1 \textbf{ or } \text{wp}(SL_2, R)) =$$
$$(B_1 \textbf{ and } \text{wp}(SL_1, R)) \textbf{ or } (\textbf{non } B_1 \textbf{ and } \text{wp}(SL_2, R)) \qquad (5)$$

로 귀착된다. 마지막 변형은 네 개의 결합된 항들[6)] 중 $B_1 \textbf{ and non } B_1 = F$라서 생략이 가능하고, 한편 $\text{wp}(SL_1, R) \textbf{ and } \text{wp}(SL_2, R)$도 또한 뺄 수 있는데, 그것이 참인 모든 상태에서 (5)의 두 항들 중 정확히 하나는 참이어야만 하므로 그 이접(離接; disjunction)에서 생략이 가능하기 때문이다. (5) 식은 씨. 에이. 알. 호아레(C. A. R. Hoare)가 ALGOL 60의 **if-then-else**에 대해 의미를 부여한 방식과 밀접하게 관련되어 있다. 여기에서 $BB = T$이고 모든 것으로부터 함축되므로,[7)] 우리는 보다 약한 가정[8)]

$$((Q \textbf{ and } B_1) \Rightarrow \text{wp}(SL_1, R)) \textbf{ and}$$
$$((Q \textbf{ and non } B_1) \Rightarrow \text{wp}(SL_2, R))$$

로도 (3)이라는 결론을 내릴 수 있다.
(주의 끝).

대안적 구조를 위한 이 정리는 술어들의 쌍인 $Q$와 $R$이 다음

$$R = P$$
$$Q = P \textbf{ and } BB$$

과 같이 기술될 수 있는 경우에 특히 중요하다. 이 경우 전제 (1)은 저절로 만족되고, 한편 전제 (2)는—$(BB \textbf{ and } B_j) = B_j$이므로—다음

3) 이 정리가 뜻하는 바를 말로 하면 다음과 같다: "$Q$가 나타내는 상태 하에서는 참인 가드가 적어도 하나는 존재하여 IF 구조가 취소(abort)되지 않고 또한 그 상태에서 참인 가드에 대응하는 문장을 수행하면 사후 조건 $R$을 만족하는 상태에서 종료한다면, $Q$를 만족하는 상태에서 IF 구조를 실행하면 $R$을 만족하는 상태에서 적절히 종료한다." 내용을 곱씹어 보면 지당한 이야기이며, $Q$와 $R$이 무관하지 않으리라는 점도 당연하다고 생각될 것이다.

4) 모두 잘 아는 것이지만 노파심에 부연하자면, 함축(implication)의 전제(antecedent)가 거짓이면 그 결론의 참 혹은 거짓에 관계없이 함축 전체는 참이 된다.

5) 앞의 각주를 참고하라.

6) 두 번째 줄의 식을 풀어 헤치면, 두 항이 공접(共接; conjunction)으로 묶여진 것들 네 쌍(pair)이 이접으로 연결되어 있다.

7) 함축에서 결론 부분이 참이면, 가정에 관계없이 그 함축은 참이므로 그러하다.

8) (2) 식을 이 경우에 맞도록 풀어 쓴 것이다. ALGOL 60의 **if-then-else** 경우 가드는 두 개이고 서로가 서로의 부정이기 때문에 이렇게 된다.

$$(\mathbf{A}\ j{:}1 \le j \le n{:}\ (P \textbf{ and } B_j) \Rightarrow \mathrm{wp}(SL_j, P)) \qquad (6)$$

으로 귀착되는데, 우리는 (3) 덕분에 이것으로부터 모든 상태에 대해

$$(P \textbf{ and } BB) \Rightarrow \mathrm{wp}(\mathrm{IF}, P) \qquad (7)$$

라는 결론을 내릴 수 있으며, 이것은 우리의 다음 정리를 위한 전제를 이루는 관계식이다.

**정리**

**반복적 구조를 위한 기본 정리.**

어떤 가드 명령 집합과 그것으로부터 만들어지는 대안적 구조 IF, 그리고 술어 $P$가 모든 상태에 대해

$$(P \textbf{ and } BB) \Rightarrow \mathrm{wp}(\mathrm{IF}, P) \qquad (7)$$

를 만족한다고 하자. 그러면 우리는 (이 가드 명령 집합에) 대응하는 반복 구조 DO에 대해, 모든 상태에서

$$(P \textbf{ and } \mathrm{wp}(\mathrm{DO}, T)) \Rightarrow \mathrm{wp}(\mathrm{DO}, P \textbf{ and non } BB) \qquad (8)$$

라는 결론을 내릴 수 있다.[9]

이 정리는 '루프를 위한 기초적 불변 정리(Fundamental Invatiance Theorem for Loops)'라고도 불리며, 이해하기에는 직관적으로 어렵지 않다. 전제 (7)은 만약 $P$가 초기에 성립하고 가드 명령들 중 하나가 실행을 위해 선택된다면, 그 수행 후 $P$가 여전히 성립함을 우리에게 말해준다. 달리 말하자면, 가드들은 $P$가 처음에 성립한다면 (가드에 대응되는) 문장 목록들의 수행이 $P$의 참을 깨뜨리지 않을 것이라는 점을 보장한다. 그 (가드 명령) 집합에 있는 어떤 가드 명령이 아무리 여러 번 선택된다 하더라도, 매번 새로이 가드들을 조사할 때마다 $P$는 따라서 성립할 것이다. 그러므로 가드들 중 어느 것도 참이 아니어서 전체 반복적 구조가 종료될 때, 우리는 $P$ **and non** $BB$를 만족하는 최종 상태에서 마칠 것이다. 질문은 이런 것이다: 그것이 적절히 종료할까? 그렇다. wp(DO, $T$)가 처음에 역시 성립한다면, 그것은 적절히 종료할 것이다; 어떤 상태이든지 $T$는 만족하므로, wp(DO, $T$)는 정의에 의해서 문장 DO를 활성화시키면 적절히 종료하는 동작으로 이끄는 초기 상태를 위한 최약 사전 조건이다.

반복적 구조를 위한 기본 정리의 정형적 증명은 그 구조의 의미를 위한 정형적 정의(앞의 장을 보라)에 기반한다. 이 정의로부터 우리는

$$H_0(T) = \textbf{non } BB \qquad (9)$$

$$k > 0\text{일 때: } H_k(T) = \mathrm{wp}(\mathrm{IF}, H_{k-1}(T)) \textbf{ or } \textbf{ non } BB \qquad (10)$$

$$H_0(P \textbf{ and non } BB) = P \textbf{ and non } BB \qquad (11)$$

$$k > 0\text{일 때: } H_k(P \textbf{ and non } BB)) = \mathrm{wp}(\mathrm{IF}, H_{k-1}(P \textbf{ and non } BB)) \textbf{ or } P \textbf{ and non } BB \qquad (12)$$

를 도출한다. 수학적 귀납법을 통해 전제 (7)이 모든 상태에 대해

$$k \ge 0\text{일 때: } (P \textbf{ and } H_k(T)) \Rightarrow H_k(P \textbf{ and non } BB) \qquad (13)$$

가 만족된다는 것이다. 즉 어떤 메커니즘(프로그램이라고 생각해도 좋을 것이다)의 일부로서 반복적 구조가 실행된다면, 그것이 실행되기 이전에 이 메커니즘의 최종적인 목표를 위해 달성된 상태를 깨뜨리지 않고 유지하면서 추가의 작업을 달성해야 한다는 뜻이다.

---

9) 이 정리를 말로 다시 쓰면 다음과 같다: "IF 구조가 취소되지 않고 실행되도록 하면서 그 실행 후에도 여전히 유지되는 조건 $P$가 있다면, $P$가 성립하고 동일 가드 명령 집합으로 이루어진 DO 구조가 적절히 종료할 때 사후 조건으로 $P$는 여전히 성립하며 추가로 **non** $BB$가 성립한다."

직관적으로 이 정리가 성립하는 것은 명백하다. 너무 명백해서 동어반복처럼 들리기까지 한다. 그러나 이 정리에서 (8) 식의 전제의 일부를 이루는 'DO 구조가 적절히 종료하는 사전 최약 조건'은 그리 간단하지 않다. (이것은 다음 장에서 자세히 다뤄질 것이다.)

이 정리에 대한 이러한 표면적 이해보다는 그것이 지향하는 바를 파악하는 것이 중요하다. 이 정리의 전제는 반복적 구조(DO)가 포함하고 있는 가드 명령 집합 중 임의의 것이 거듭 실행된다고 하더라도 변함 없이 유지되는 상태($P$)가 있다는 것이며, 그 결과 반복적 구조가 종료될 때는 참인 가드가 하나도 없다는 뜻이므로 이 유지되는 상태($P$)에 더하여 추가 상태(**non** $BB$)

를 보장한다는 것을 보임으로써 시작한다.

관계식 (9)와 (11)은 (13)이 $k = 0$에 대해 성립함을 말해준다. 우리는 (13)이 $k = K-1$에 대해 성립한다는 가정 하에 $k = K(K>0)$에 대해 (13)이 성립될 수 있음을 보이고자 한다.

$$\begin{aligned} P \text{ and } H_k(T) &= P \text{ and } \mathrm{wp}(\mathrm{IF}, H_{K-1}(T)) \text{ or} \\ &\quad\ P \text{ and non } BB \\ &= P \text{ and } BB \text{ and } \mathrm{wp}(\mathrm{IF}, H_{K-1}(T)) \text{ or} \\ &\quad\ P \text{ and non } BB \\ &\Rightarrow \mathrm{wp}(\mathrm{IF}, P) \text{ and } \mathrm{wp}(\mathrm{IF}, H_{K-1}(T)) \text{ or} \\ &\quad\ P \text{ and non } BB \\ &= \mathrm{wp}(\mathrm{IF}, P \text{ and } H_{K-1}(T)) \text{ or} \\ &\quad\ P \text{ and non } BB \\ &\Rightarrow \mathrm{wp}(\mathrm{IF}, H_{K-1}(P \text{ and non } BB)) \text{ or} \\ &\quad\ P \text{ and non } BB \\ &= H_K(P \text{ and non } BB) \end{aligned}$$

첫 번째 줄의 동치는 (10)에서 연유하고, 두 번째 줄의 동치는 어떤 wp(IF, $R$)이든지 $BB$를 함축한다는 사실에서 연유하며,[10] 세 번째 줄의 함축은 (7)로부터 연유하고, 네 번째 줄의 동치는 술어 변환자를 위한 성질 3에서 연유하며,[11] 다섯 번째 줄의 함축은 술어 변환자를 위한 성질 2와 $k = K-1$에 대해 가정한 (13)으로부터 연유하고, 마지막 줄은 (12)에서 연유한다. 그래서 이제 (13)은 $k = K$에 대해서 증명이 되었으며, 따라서 모든 $k \geq 0$에 대해 성립한다.

마지막으로, 우리는—(13) 덕분에—상태 공간 내의 임의의 점에 대해

$$\begin{aligned} P \text{ and } \mathrm{wp}(\mathrm{DO}, T) &= (\,\mathbf{E}\ k{:}\ k \geq 0{:}\ P \text{ and } H_k(T)) \\ &\Rightarrow (\,\mathbf{E}\ k{:}\ k \geq 0{:}\ H_k(P \text{ and non } BB)) \\ &= \mathrm{wp}(\mathrm{DO}, P \text{ and non } BB) \end{aligned}$$

이고, 따라서 반복적 구조를 위한 기본 정리 (8)이 증명되었다. 반복적 구조를 위한 기본 정리의 최대 유용성은 그 전제나 결론 어디에도 가드 명령이 선택되는 실제 횟수에 대해 언급하지 않는다는 사실로부터 나온다. 결과적으로, 이것 덕분에 초기 상태에 의해 그 (반복) 횟수가 정해지지 않는 경우에서조차 여러 단언(斷言; assertion)[12]이 가능하다.

10) wp(IF, $R$)의 정형적 의미는 IF 구조의 취소를 방지하기 위해 공접의 한 항으로서 $BB$를 포함하고 있다. (앞의 장에 나왔다.) 따라서 $BB$를 공접에 추가시켜도 그 진리값에 변화는 없다.

11) 다익스트라의 책의 제3장에 나온다.

12) 예전 강의에서 설명한 바가 있듯이, 프로그램의 상태에 대한 지정을 의미한다.

## 참고 문헌


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

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

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

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

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

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

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

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

## 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:
뉴욕주립대학교 컴퓨터과학과 객원교수

관심분야:
프로그래밍 언어