GLAVA 4

Univerzalni program

Nesavršenstvo nije negacija savršenstva.
To se samo cjelina iskazuje u djelovima,
to se beskraj otkriva u granicama.
Rabindranat Tagore

3. Problem završetka rada programa (HALTING)

Sada ćemo razmatrati predikat HALT(x,y), koji se definiše na sljedeći način.

Neka je, za neko dato y, Π program takav da je #(Π)=y. Tada kažemo da je HALT(x,y) istinit ako je $\Psi_{\Pi}^{(1)}$(x) definisano, a neistinit ako je $\Psi_{\Pi}^{(1)}$(x) nedefinisano. Drugim riječima:

HALT(x,y) $\Leftrightarrow$ program čiji je proj y se zaustavlja za ulaz x.

Sada možemo dokazati značajnu teoremu.

Teorema 3.1. HALT(x,y) nije izračunljiv predikat.

Dokaz. Pretpostavimo da je HALT(x,y) izračunljiv. Sada možemo napraviti program Π:

[A] IF HALT(x,y) GOTO A

Sasvim je jasno da je Π program za koji važi:

$$\Psi_{\Pi}^{(1)}(x) = \begin{cases} nedefinisano, & ako\ je \quad HALT(x,y) \\ 0, & ako\ je \sim HALT(x,y) \end{cases}$$

Neka je #(Π) = y_0. Tada iz definicije HALT redikata slijedi:

HALT(x, y_0) $\Leftrightarrow \sim$ HALT(x,x).

Pošto gornja ekvivalencija važi za svako x, važi i za x = y_0:

HALT(y_0, y_0) $\Leftrightarrow \sim$ HALT(y_0,y_0).

Ovo je kontradikcija koja dokazuje teoremu.

Ova teorema nam je dala primjer funkcije koja nije izračunljiva nijednim programom napisanim u jeziku S. Mi bi pokušali da zaključimo još opštije:

Ne postoji algoritam koji za dati program u jeziku S, i za dati ulaz može odrediti da li će dati program zaustaviti rad posle konačnog broja izvršavanja instrukcija.

Ovako izrečen rezultat se naziva **nerešivost problema zustavljanja**. Možemo rezonovati na sljedeći način: Ako postoji takav algoritam, možemo ga koristiti za provjeru istinitosti predikata HALT(x,y) za date x i y, tako što najprije dekodiramo program Δ takav da je $\#(\Delta) = y$ i provjerimo da li se Δ zaustavlja pri ulazu x. Međutim, imamo razloga da vjerujemo da **bilo koji algoritam za izračunavanje sa brojevima može biti iskazan u jeziku S**. Zato bi ovakvo rezonovanje bilo u kontradikciji sa već dokazanom teoremom da HALT(x,y) nije izračunljiv predikat.

Vjerovanje da "bilo koji algoritam za izračunavanje sa brojevima može biti iskazan u jeziku S", naziva se **Church-ovom tezom**. Za takvo vjerovanje smo već dali dosta argumenata. Ali kako izraz "algoritam" nema definiciju koja bi bila nezavisna od programskog jezika kojim se algoritam izražava, Church-ova teza ne može biti dokazana kao matematička teorema.

Mi ćemo Church-ovu tezu koristiti slobodno i govoriti o nepostojanju algoritma kada god pokažemo da se problem ne može rijesiti jezikom S.

Uz Church-ovu tezu, Teorema 3.1. zapravo kaže da ne postoji algoritam kojim se testira da li dati program, za dati ulaz, ikada zaustavlja rad. Za one kojima ovaj rezultat izgleda iznenađujući, za tako "jednostavan" problem, daćemo jedan primjer programa za koji se ne zna da li zaustavlja rad.

Jednostvano je napisati program koji "rješava" sljedeći problem.
Odrediti bar jedno rješenje jednačine:

$$x^n + y^n = z^n \qquad \text{za } n > 2. \qquad (3.1)$$

Ovaj problem je poznat kao Fermat-ova posljednja teorema. Uprkos izrazu teorema ovaj 300 godina star problem je još uvijek otvoren. Pitanje da li jednačina (3.1.) ima rješenje je ekvivalentno pitanju da li program, koji traži takvo rješenje, ikada završava rad.

Vježbe

1. Pokazati da HALT(x,x) nije izračunljiv predikat.