Κεφάλαιο 5
Αξιωματική Σημασιολογία και Απόδειξη Ορθότητας Προγραμμάτων

Π. Ροντογιάννης

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Εισαγωγή

- Τα προγράμματα μιας (κλασικής) γλώσσας προγραμματισμού αποτελούνται από εντολές (ανάθεσης, επανάληψης κλπ)
- Βασικός σκοπός του κεφαλαίου αυτού είναι η μελέτη της τεχνικής των Floyd-Hoare για απόδειξη ορθότητας τέτοιων προγραμμάτων.

Κανόνες

Το συντακτικό που θα εξετασθεί είναι το:

Συντακτικό

$$
\begin{array}{rcl}
C & ::= & \texttt{skip} \\
 & | & \texttt{(}\ C\ \texttt{)} \\
 & | & i := E \\
 & | & C_0\ \texttt{;}\ C_1 \\
 & | & \texttt{while}\ B\ \texttt{do}\ C \\
 & | & \texttt{if}\ B\ \texttt{then}\ C_0\ \texttt{else}\ C_1
\end{array}
$$

Τεχνική Floyd-Hoare

- Ο C.A.R. Hoare εισήγαγε τον συμβολισμό για τον καθορισμό του τι κάνει ένα πρόγραμμα:

$$\{P\}C\{Q\}$$

- όπου C είναι το πρόγραμμα της γλώσσας που μελετάται
- και P και Q συνθήκες, που σχετίζονται με μεταβλητές που χρησιμοποιεί το C

Συνθήκη

- Αποτελεί έκφραση μιας απλής λογικής γλώσσας
  - Περιέχει μεταβλητές του προγράμματος, σταθερές, λογικούς τελεστές κλπ
- Όταν προηγείται μιας εντολής περιγράφει περιορισμούς για μεταβλητές του προγράμματος σε εκείνο το σημείο
- Όταν ακολουθεί εντολή περιγράφει τους νέους περιορισμούς για τις μεταβλητές, μετά την εκτέλεση της εντολής
- Εκφράσεις της μορφής $\{P\}C\{Q\}$ ονομάζονται **προδιαγραφές** (specifications)

Ορισμός 5.1

Ορισμός

Μία προδιαγραφή $\{P\}C\{Q\}$ είναι αληθής
εάν όταν το C εκτελείται σε μία κατάσταση που ικανοποιεί
τη συνθήκη P και αν η εκτέλεση του C τερματίζει,
τότε στην κατάσταση στην οποία καταλήγει το πρόγραμμα,
ικανοποιείται η συνθήκη Q

Ορισμός 5.1

- Οι προδιαγραφές ονομάζονται και **τύποι μερικής ορθότητας** διότι η απόδειξη του ότι ένα πρόγραμμα τερματίζει θεωρείται ξεχωριστή διαδικασία, που πρέπει να εξετάσει ο προγραμματιστής, για να είναι βέβαιος για την ορθότητα του προγράμματος του.

Παράδειγμα 5.1

Η προδιαγραφή:

$$\{X = 1\}\ \texttt{X:= X + 1}\ \{X = 2\}$$

είναι αληθής.

(Το σύμβολο $=$ είναι η γνωστή μας μαθηματική ισότητα).

Παράδειγμα 5.2

Η προδιαγραφή:

$$\{X = 1\} \ \texttt{Y:= X} \ \{Y = 1\}$$

είναι αληθής.

Παράδειγμα 5.3

Η προδιαγραφή:

$$\{X = 1\} \ \texttt{Y:= X + 1} \ \{X = 2\}$$

είναι ψευδής.

Παράδειγμα 5.4

Η προδιαγραφή:

$$\Big\{(X = x) \wedge (Y = y)\Big\}\ \texttt{R:=X}\ ;\ \texttt{X:=Y}\ ;\ \texttt{Y:=R}\ \Big\{(X = y) \wedge (Y = x)\Big\}$$

είναι αληθής.

Οι μεταβλητές x και y, που εμφανίζονται στις συνθήκες, αλλά όχι στην εντολή, ονομάζονται **βοηθητικές μεταβλητές**. Σκοπός τους είναι να δώσουν όνομα στις αρχικές τιμές των X και Y αντίστοιχα.

Παράδειγμα 5.5

Η προδιαγραφή:

$$\{true\}C\{Q\}$$

Είναι αληθής, εάν όταν το πρόγραμμα C τερματίζει, ισχύει η συνθήκη Q.

Παράδειγμα 5.6

Η προδιαγραφή:

$$\{P\}C\{true\}$$

είναι πάντα αληθής για κάθε αρχική συνθήκη P και κάθε εντολή C.

Τεχνική Floyd-Hoare

- Δημιουργία τυπικών αποδείξεων για προδιαγραφές προγραμμάτων
  - Χρήση αξιωμάτων και κανόνων εξαγωγής συμπερασμάτων, που υποστηρίζει η τεχνική
  - Χρήση θεωρημάτων κλασσικών μαθηματικών

Διαφορά Αξιωμάτων και Κανόνων

- **Αξίωμα** είναι μία λογική πρόταση για την οποία υποθέτουμε ότι είναι αληθής
- **Κανόνας εξαγωγής συμπερασμάτων** είναι μία μέθοδος, που χρησιμοποιείται για να δείξουμε ότι μία προδιαγραφή είναι αληθής με την υπόθεση ότι άλλες προδιαγραφές είναι αληθείς

Αξίωμα του skip

Η εντολή skip δε μεταβάλλει οποιαδήποτε συνθήκη ισχύει πριν από την εκτέλεση της

Αξίωμα

$$\{P\}\ \texttt{skip}\ \{P\}$$

Κανόνας των παρενθέσεων

Κανόνας

Εάν ισχύει το $\{P\}C\{R\}$ *τότε μπορούμε να εξάγουμε ως συμπέρασμα και το*

$$\{P\} \ (\ C \ ) \ \{R\}$$

Αξίωμα Εντολής Ανάθεσης

Αναπαριστά το γεγονός ότι η τιμή μιας μεταβλητής `V` μετά από την εκτέλεση της εντολής ανάθεσης `V:=E` ισούται με την τιμή της έκφρασης E στην κατάσταση πριν την εκτέλεση της εντολής.

Αξίωμα Εντολής Ανάθεσης

Αξίωμα

$$\{P[E/V]\}\ \mathtt{V}\ :=\ \mathtt{E}\ \{P\}$$

όπου V μεταβλητή, E έκφραση, P συνθήκη και $P[E/V]$ το αποτέλεσμα της αντικατάστασης όλων των εμφανίσεων της μεταβλητής V στην P με E.

Παράδειγμα 5.7

Εφαρμογή του αξιώματος της Εντολής Ανάθεσης

$$\{X + 1 = n + 1\}\ \texttt{X := X+1}\ \{X = n + 1\}$$

Όπως και

$$\{E = E\}\ \texttt{X:= E}\ \{X = E\}$$

αν το X δεν εμφανίζεται στο E.

Μορφή

- Μία συχνή παρανόηση είναι ότι το αξίωμα Ανάθεσης θα έπρεπε να είναι: $\{P\}$ `V:=E` $\{P[E/V]\}$
- Είναι όμως λανθασμένο, γιατί θα έδινε προδιαγραφές: $\{X = 0\}$ `X:=1` $\{1 = 0\}$
- Το (σωστό) αξίωμα Ανάθεσης δεν ισχύει (σε αυτή τη μορφή) για πιο πολύπλοκες προστακτικές γλώσσες προγραμματισμού.

Κανόνας Ενδυνάμωσης της Προσυνθήκης

Κανόνας

Εάν ισχύει το $P \Rightarrow R$ καθώς και το $\{R\}C\{Q\}$ τότε μπορούμε να εξάγουμε ως συμπέρασμα και το $\{P\}C\{Q\}$.

Παράδειγμα 5.8

$$\{X + 1 = n + 1\} \ \texttt{X} \ := \ \texttt{X+1} \ \{X = n + 1\}$$

$$\text{και} \ \ X = n \Longrightarrow X + 1 = n + 1$$

Επομένως με βάση τον Κανόνα της Ενδυνάμωσης της Προσυνθήκης, εξάγουμε ως συμπέρασμα

$$\{X = n\} \ \texttt{X} \ := \ \texttt{X+1} \ \{X = n + 1\}$$

Η n είναι η βοηθητική μεταβλητή για συσχετισμό τιμών στην κατάσταση πριν και μετά την εκτέλεση εντολής.

Κανόνας Αποδυνάμωσης της Μετασυνθήκης

Κανόνας

Εάν ισχύει το $\{P\}C\{R\}$ *καθώς και το* $R \Rightarrow Q$ *τότε μπορούμε να εξάγουμε ως συμπέρασμα και το* $\{P\}C\{Q\}$.

Παράδειγμα 5.9

Με χρήση του αξιώματος της Εντολής Ανάθεσης έχουμε:

$$\Big\{(R = X) \wedge (0 = 0)\Big\} \ \texttt{Q:=0} \ \Big\{(R = X) \wedge (Q = 0)\Big\}$$

Από τα κλασσικά μαθηματικά γνωρίζουμε

$$R = X \Longrightarrow (R = X) \wedge (0 = 0)$$

Με χρήση του κανόνα Ενδυνάμωσης της Προσυνθήκης έχουμε

$$\Big\{R = X\Big\} \ \texttt{Q:=0} \ \Big\{(R = X) \wedge (Q = 0)\Big\}$$

Παράδειγμα 5.9

Με χρήση πάλι απλών μαθηματικών έχουμε

$$(R = X) \wedge (Q = 0) \Longrightarrow R = X + YQ$$

Με χρήση του κανόνα Αποδυνάμωσης της Μετασυνθήκης έχουμε

$$\left\{R = X\right\} \ \mathtt{Q:=0} \ \left\{R = X + YQ\right\}$$

Οι κανόνες Ενδυνάμωσης και Αποδυνάμωσης ονομάζονται και **κανόνες Συνεπαγωγής**.

Κανόνες Σύζευξης και Διάζευξης

Κανόνας

Εάν ισχύει το $\{P_1\}C\{Q_1\}$ *καθώς και το* $\{P_2\}C\{Q_2\}$ *τότε μπορούμε να εξάγουμε ως συμπέρασμα και το*

$$\{P_1 \wedge P_2\}C\{Q_1 \wedge Q_2\}$$

Κανόνας

Εάν ισχύει το $\{P_1\}C\{Q_1\}$ *καθώς και το* $\{P_2\}C\{Q_2\}$ *τότε μπορούμε να εξάγουμε ως συμπέρασμα και το*

$$\{P_1 \vee P_2\}C\{Q_1 \vee Q_2\}$$

Κανόνας Σύνθετων Εντολών

Αφορά εντολές της μορφής $C_1 ; C_2$.

Κανόνας

Εάν ισχύουν τα $\{P\}C_1\{Q\}$ *και* $\{Q\}C_2\{R\}$ *μπορούμε να εξάγουμε το συμπέρασμα*

$$\{P\}C_1;C_2\{R\}$$

Παράδειγμα 5.10

Με χρήση του αξιώματος της εντολής ανάθεσης μπορούμε να εξάγουμε ότι ισχύουν τα ακόλουθα:

$$\{(X = x) \wedge (Y = y)\}\ \texttt{R:= X}\ \{(R = x) \wedge (Y = y)\}$$
$$\{(R = x) \wedge (Y = y)\}\ \texttt{X:= Y}\ \{(R = x) \wedge (X = y)\}$$
$$\{(R = x) \wedge (X = y)\}\ \texttt{Y:= R}\ \{(Y = x) \wedge (X = y)\}$$

Παράδειγμα 5.10

Από τα δύο πρώτα και με βάση τον κανόνα των Σύνθετων Εντολών μπορούμε να εξάγουμε ότι ισχύουν τα ακόλουθα:

$$\Big\{(X = x) \wedge (Y = y)\Big\}\ \texttt{R:=X}\ ;\ \texttt{X:=Y}\ \Big\{(R = x) \wedge (X = y)\Big\}$$

Τα παραπάνω δίνουν

$$\Big\{(X = x) \wedge (Y = y)\Big\}\ \texttt{R:=X}\ ;\ \texttt{X:=Y}\ ;\ \texttt{Y:=R}\ \Big\{(X = y) \wedge (Y = x)\Big\}$$

Κανόνας του if

Κανόνας

Εάν ισχύει $\{P \wedge S\} C_1 \{Q\}$ *καθώς και* $\{P \wedge \neg S\} C_2 \{Q\}$ *τότε μπορούμε να εξάγουμε ως συμπέρασμα το*

$$\{P\} \texttt{ if } S \texttt{ then } C_1 \texttt{ else } C_2 \{Q\}$$

Παράδειγμα 5.11

Με χρήση του κανόνα του `if` καθώς και άλλους προηγούμενους κανόνες και αξιώματα η προδιαγραφή

$$\{y > 1\}\ \texttt{if (x>0) then y:=y-1 else y:=y+1}\ \{y > 0\}$$

είναι αληθής.

Κανόνας του while

Κανόνας

Αν γνωρίζουμε ότι $\{P \wedge S\}C\{P\}$ *τότε μπορούμε να εξάγουμε ως συμπέρασμα ότι*

$$\{P\} \texttt{ while } S \texttt{ do } C \ \{P \wedge \neg S\}$$

Η συνθήκη P ονομάζεται **αμετάβλητη συνθήκη** (invariant) διότι εξακολουθεί και ισχύει και μετά το τέλος της εκτέλεσης της εντολής `while`.

Κανόνας του while - 2

Ο κανόνας του while λέει ότι εάν P είναι μία αμετάβλητη συνθήκη του σώματος μιας εντολής while (όταν ισχύει και η συνθήκη S) τότε η P είναι μία αμετάβλητη συνθήκη ολόκληρης της εντολής while.

Παράδειγμα 5.12

Έστω το πρόγραμμα $\mathcal{P}$:

Πρόγραμμα

```
R:=X;
Q:=0;
while (Y<= R) do (R:=R-Y ; Q:=Q+1)
```

Θέλουμε να δείξουμε ότι ο αλγόριθμος υπολογίζει το πηλίκο Y και το υπόλοιπο R της διαίρεσης του X με το Y, δηλαδή:

$$\Big\{(Y > 0) \wedge (X \geq 0)\Big\}\ \mathcal{P}\ \Big\{(R < Y) \wedge (X = R + YQ)\Big\}$$

Ερώτηση: Γιατί παίρνουμε $Y > 0$;

Παράδειγμα 5.12

Απόδειξη:

Πρώτα θα δείξουμε ότι:

$$\{(Y > 0) \wedge (X \geq 0)\} \texttt{ R:=X; Q:=0 } \{X = R + YQ\}$$

και μετά

$$\Big\{X = R + YQ\Big\}$$

```
while (Y<=R) do ( R:=R-Y ; Q:=Q+1 )
```

$$\Big\{(X = R + YQ) \wedge \neg(Y \leq R)\Big\}$$

Παράδειγμα 5.12

Η πρώτη αποδεικνύεται εύκολα. Για τη δεύτερη θα δείξουμε πρώτα ότι:

$$\{X = R + YQ\}\ \texttt{R:=R-Y}\ ;\ \texttt{Q:=Q+1}\ \{X = R + YQ\}$$

Με βάση το Αξίωμα Εντολής Ανάθεσης

$$\{X = (R - Y) + Y + YQ\}\ \texttt{R:=R-Y}\ \{X = R + Y + YQ\}$$

$$\{X = R + Y(Q + 1)\}\ \texttt{Q:=Q+1}\ \{X = R + YQ\}$$

Με χρήση του Κανόνα Σύνθετων Εντολών

$$\{X = R + YQ\}\ \texttt{R:=R-Y}\ ;\ \texttt{Q:=Q+1}\ \{X = R + YQ\}$$

Παράδειγμα 5.12

Με χρήση του Κανόνα Ενδυνάμωσης της Προσυνθήκης

$$\Big\{(X = R + YQ) \wedge (Y \leq R)\Big\}$$

`R:=R-Y ; Q:=Q+1`

$$\Big\{X = R + YQ\Big\}$$

Με εφαρμογή του Κανόνα του `while` θα πάρουμε το ζητούμενο αποτέλεσμα.

Παράδειγμα

Έστω το πρόγραμμα:

Πρόγραμμα

```
j:=0;
i:=k;
while (i<n) do
  j:=j+1;
  i:=i+1;
end
```

Βρείτε μια αμετάβλητη συνθήκη για το while loop.

Λύση

Δοκιμάζουμε για μερικές επαναλήψεις:

i	j
k	0
$k+1$	1
$k+2$	2
$k+3$	3
$\ldots$	$\ldots$

Λύση

Προφανώς το $i - j = k$ είναι μια αμετάβλητη συνθήκη. Για να το δείξουμε και τυπικά, αρκεί να δείξουμε ότι

$$\Big\{(i - j = k) \wedge (i < n)\Big\}\ \texttt{j:=j+1}\ \ ;\ \ \texttt{i:=i+1}\ \Big\{i - j = k\Big\}$$

Αρχίζοντας από το τέλος έχουμε:

$$\{i + 1 - j = k\}\ \texttt{i:=i+1}\ \{i - j = k\}$$

$$\{i + 1 - j - 1 = k\}\ \texttt{j:=j+1}\ \{i + 1 - j = k\}$$

Άρα το $i - j = k$ είναι μια αμετάβλητη συνθήκη.

Παράδειγμα

Έστω το πρόγραμμα $\mathcal{P}$:

Πρόγραμμα

```
K:=N;
s:=1;
while (K>0) do
  s:=A*s;
  K:=K-1;
end
```

Θέλουμε να δείξουμε ότι

$$\left\{(N > 0) \wedge (A \geq 0)\right\} \mathcal{P} \left\{s = A^N\right\}$$

Λύση

- Πώς βρίσκουμε τις αμετάβλητες συνθήκες;
- Δοκιμάζουμε τον κώδικα για μικρούς αριθμούς π.χ. $A = 2,\ N = 5$:

K	s	sA^K
5	1	32
4	2	32
3	4	32
2	8	32
1	16	32
0	32	32

Λύση

- Μια αμετάβλητη συνθήκη που θα μπορούσαμε να χρησιμοποιήσουμε είναι η

$$sA^K = A^N$$

- Βλέπουμε επίσης ότι μία άλλη αμετάβλητη συνθήκη φαίνεται να είναι η $K \geq 0$, διότι

$$\left\{(K \geq 0) \wedge (K > 0)\right\} \texttt{s:=A*s ; K:=K-1} \left\{K \geq 0\right\}$$

- Επομένως διαλέγουμε ως αμετάβλητη συνθήκη τη συνθήκη:

$$(sA^K = A^N) \wedge (K \geq 0)$$

Λύση

Δείχνουμε καταρχήν ότι:

$$\left\{(sA^K = A^N) \wedge (K \geq 0) \wedge (K > 0)\right\}$$

$$\texttt{s:=A*s ; K:=K-1} \left\{(sA^K = A^N) \wedge (K \geq 0)\right\}$$

ή ισοδύναμα

$$\left\{(sA^K = A^N) \wedge (K > 0)\right\} \texttt{s:=A*s ; K:=K-1} \left\{(sA^K = A^N) \wedge (K \geq 0)\right\}$$

Λύση

Αρχίζοντας από το τέλος έχουμε:

$$\left\{(sA^{K-1} = A^N) \wedge (K-1 \geq 0)\right\} \texttt{K:=K-1} \left\{(sA^K = A^N) \wedge (K \geq 0)\right\}$$

$$\left\{(AsA^{K-1} = A^N) \wedge (K-1 \geq 0)\right\} \texttt{s:=A*s} \left\{(sA^{K-1} = A^N) \wedge (K-1 \geq 0)\right\}$$

Είναι $(sA^K = A^N) \wedge (K > 0) \Longrightarrow (AsA^{K-1} = A^N) \wedge (K-1 \geq 0)$ άρα

$$\left\{(sA^K = A^N) \wedge (K > 0)\right\} \texttt{s:=A*s ; K:=K-1} \left\{(sA^K = A^N) \wedge (K \geq 0)\right\}$$

Λύση

Επομένως για όλο το while έχουμε:

$$\left\{(sA^K = A^N) \wedge (K \geq 0)\right\} \texttt{while...end} \left\{(sA^K = A^N) \wedge (K \geq 0) \wedge \neg(K > 0)\right\}$$

Όμως $(sA^K = A^N) \wedge (K \geq 0) \wedge \neg(K > 0) \Longrightarrow (sA^K = A^N) \wedge (K = 0)$
$\Longrightarrow s = A^N$ άρα

$$\left\{(sA^K = A^N) \wedge (K \geq 0)\right\} \texttt{while...end} \left\{s = A^N\right\}$$

Λύση

Μένει να δείξουμε ότι

$$\Big\{(N > 0) \wedge (A \geq 0)\Big\}\ \texttt{K:=N}\ ;\ \texttt{s:=1}\ \Big\{(sA^K = A^N) \wedge (K \geq 0)\Big\}$$

Αρχίζοντας από το τέλος έχουμε

$$\Big\{(A^K = A^N) \wedge (K \geq 0)\Big\}\ \texttt{s:=1}\ \Big\{(sA^K = A^N) \wedge (K \geq 0)\Big\}$$

$$\Big\{(A^N = A^N) \wedge (N \geq 0)\Big\}\ \texttt{K:=N}\ \Big\{(A^K = A^N) \wedge (K \geq 0)\Big\}$$

Όμως έχουμε $(N > 0) \wedge (A \geq 0) \Longrightarrow (N \geq 0) \Longrightarrow \texttt{true} \wedge (N \geq 0)$ $\Longrightarrow (A^N = A^N) \wedge (N \geq 0)$ επομένως το ζητούμενο ισχύει.

Παράδειγμα

Έστω το πρόγραμμα $\mathcal{P}$:

Πρόγραμμα

```
count:=n;
fact:=1;
while (count != 0) do
  fact:=fact*count;
  count:=count-1;
end
```

Θέλουμε να δείξουμε ότι

$$\left\{ n \geq 0 \right\} \mathcal{P} \left\{ fact = n! \right\}$$

Λύση

- Δοκιμάζουμε μερικές τιμές:

count	*fact*
n	1
$n-1$	n
$n-2$	$(n-1)n$
$n-3$	$(n-2)(n-1)n$

- Γενικά είναι $count! \cdot fact = n!$

Λύση

- Το παραπάνω όταν τελειώσει το loop δίνει

$$0! \cdot fact = n!$$

που είναι αυτό που θέλουμε. Διαλέγουμε λοιπόν ως αμετάβλητη συνθήκη το:

$$(count! \cdot fact = n!) \wedge (count \geq 0)$$

- Μένει να εξετάσουμε αν είναι αμετάβλητη συνθήκη.

Λύση

Αρχικά δείχνουμε ότι

$$\Big\{(count! \cdot fact = n!) \wedge (count \geq 0) \wedge (count \neq 0))\Big\}$$

```
fact:=fact*count; count:=count-1;
```

$$\Big\{(count! \cdot fact = n!) \wedge (count \geq 0)\Big\}$$

Αρχίζοντας από το τέλος έχουμε

$$\Big\{((count - 1)! \cdot fact = n!) \wedge (count \geq 1)\Big\}$$

```
count:=count-1;
```
$$\Big\{(count! \cdot fact = n!) \wedge (count \geq 0)\Big\}$$

Λύση

και στη συνέχεια

$$\Big\{((count-1)! \cdot fact \cdot count = n!) \wedge (count \geq 1)\Big\}$$

$$\texttt{fact:=fact*count;} \quad \Big\{((count-1)! \cdot fact = n!) \wedge (count \geq 1)\Big\}$$

Όμως ισχύει ότι

$$(count! \cdot fact = n!) \wedge (count \geq 0) \wedge (count \neq 0)) \Longrightarrow$$

$$\Longrightarrow (count \cdot (count-1)! \cdot fact = n!) \wedge (count \geq 1)$$

Άρα η $(count! \cdot fact = n!) \wedge (count \geq 0)$ είναι αμετάβλητη συνθήκη.

Λύση

Επομένως για όλο το while έχουμε:

$$\Big\{(count! \cdot fact = n!) \wedge (count \geq 0)\Big\} \texttt{ while...end}$$

$$\Big\{(count! \cdot fact = n!) \wedge (count \geq 0) \wedge (count = 0)\Big\}$$

Όμως $(count! \cdot fact = n!) \wedge (count \geq 0) \wedge (count = 0) \Longrightarrow fact = n!$ άρα

$$\Big\{(count! \cdot fact = n!) \wedge (count \geq 0)\Big\} \texttt{ while...end } \Big\{fact = n!\Big\}$$

Λύση

Μένει να δείξουμε ότι

$$\left\{n \geq 0\right\} \texttt{ count:=n; fact:=1 } \left\{(count! \cdot fact = n!) \wedge (count \geq 0)\right\}$$

Το οποίο είναι απλό

$$\left\{(count! = n!) \wedge (count \geq 0)\right\} \texttt{ fact:=1 } \left\{(count! \cdot fact = n!) \wedge (count \geq 0)\right\}$$

$$\left\{(n! = n!) \wedge (n \geq 0)\right\} \texttt{ count:=n; } \left\{(count! = n!) \wedge (count \geq 0)\right\}$$