ВЕРИФИКАЦИЯ ПРОГРАММ

ДВС

Лектор - С.А. Ивановский

Лекция 5

1. О схемах программ
2. Схема проектирования цикла с помощью инварианта

Схема программы

(Котов В.Е., Сабельфельд В.К. Теория схем программ. М.: Наука, 1991 – 248 с.)

Пример

```
// вычисление n!
p = 1; k = n;
while (k ≠ 0) {
   p = k * p;
   k = k – 1;
}
```

Ручная прокрутка программы при n=5

номер шага	p	k
0	1	5
1	1*5	4
2	5*4	3
3	5*4*3	2
4	5*4*3*2	1
5	5*4*3*2*1	0

Схема программы

```
// вычисление n! (n≥0)
p = 1; k = n;
while (k !=0) {
   p = k *p;
   k = k – 1;
}
```

```
// Схема программы:
p = pp; k = n;
while (k !=kk) {
   p = h (k, p);
   k = g (k);
}
```

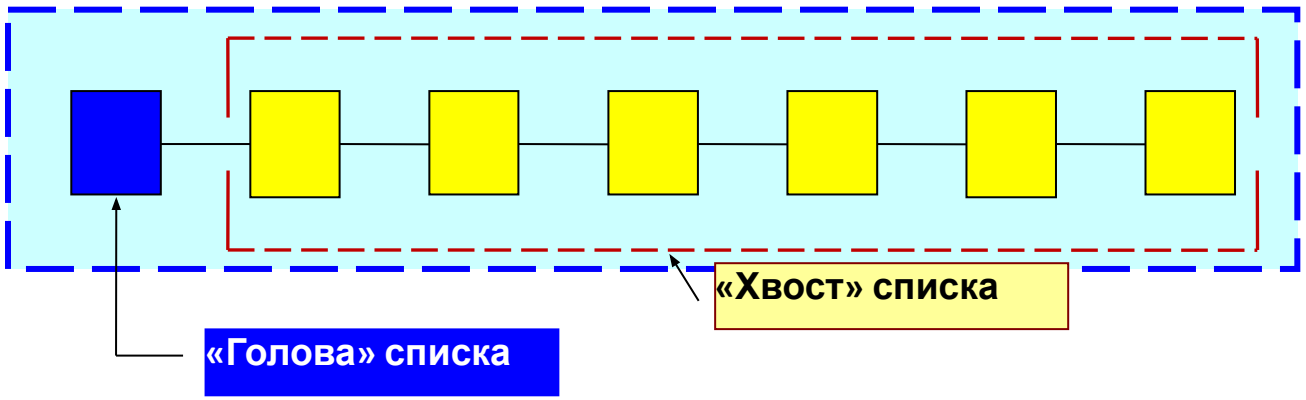
Интерпретация схемы:

pp = 1; *kk* = 0; // константы

h(*x*, *y*) = *x* * *y*; *g*(*x*) = *x* – 1;

Списки (напоминание)

Модельное представление линейного списка



Скобочная запись:

x = (**a b c d e**) или [**a, b, c, d, e**] или (**a; b; c; d; e**)

Функции-селекторы: *«голова»* - ***head***, *first*

«хвост» - ***tail***, *rest*

применяются к *непустому списку* и позволяют разобрать его на составные части.

Например, *Head* (*x*) = **a**, *Tail* (*x*) = (**b c d e**),

Head (*Tail* (*x*)) = **b**, *Tail* (*Tail* (*x*)) = (**c d e**)

Функция-конструктор ***Cons (x, y)***

x = **a** – элемент базового типа,

y = (**b c d e**) – список

Cons (*x*, *y*) = (**a b c d e**)

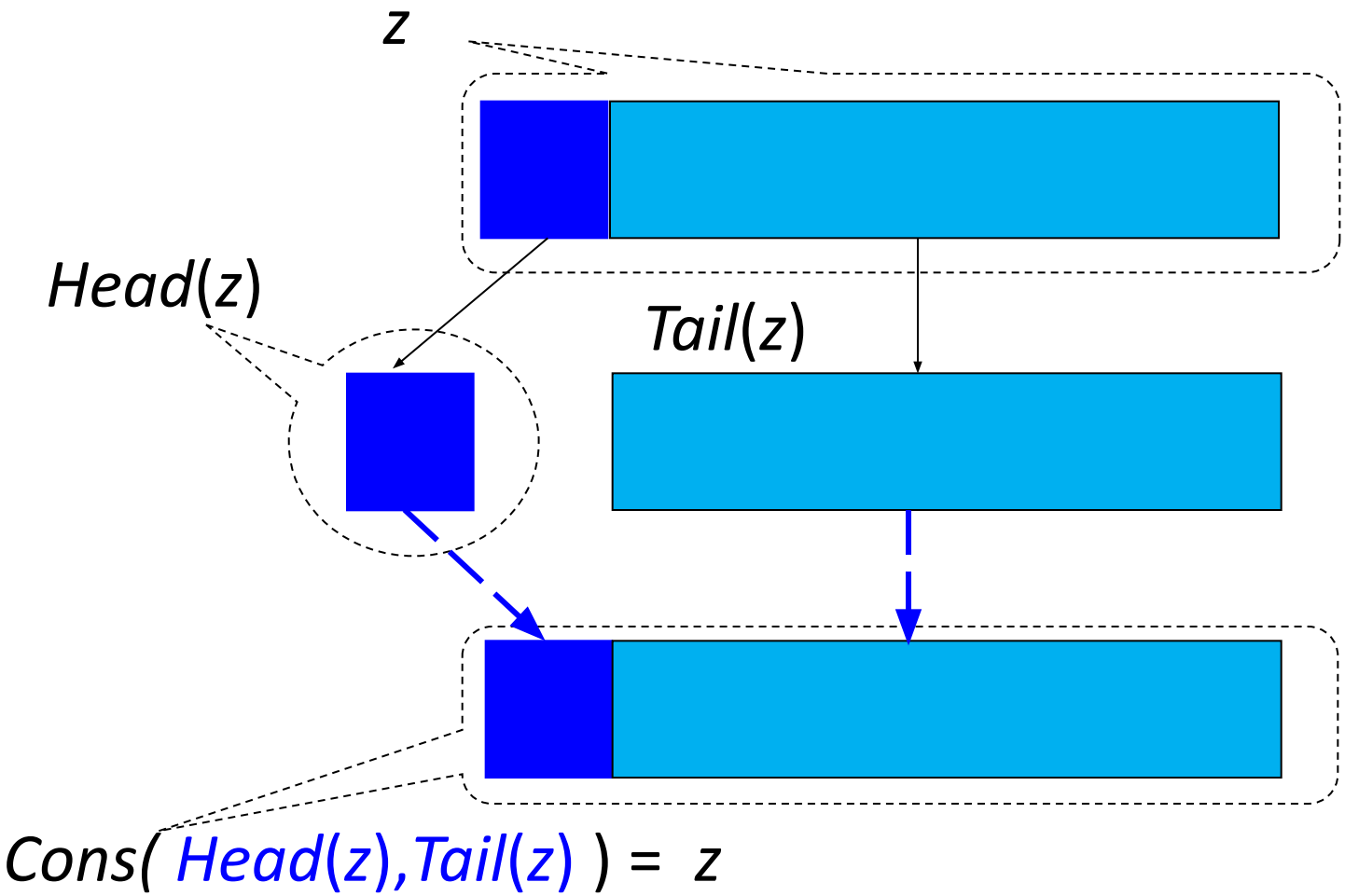
Свойства:

Cons (*Head* (*z*), *Tail* (*z*)) = *z*

Head(*Cons* (*x*, *y*)) = *x*

Tail (*Cons* (*x*, *y*)) = *y*

Иллюстрация
Cons (Head (z), Tail (z)) = z



Функция-конструктор **Cons(x, y)**

Пустой список : ( ) или **Nil**

Cons (**a**, **Nil**) = *Cons* (**a**, ( )) = (**a**)

(**a b c d e**) =

Cons (**a**, *Cons* (**b**, *Cons* (**c**, *Cons* (**d**, *Cons*(**e**, **Nil**)))))

Это «операционное» представление списка

Функция-индикатор: *Null* (z)

Null (**Nil**) = **true**, z = (**a b c d e**) → *Null* (z) = **false**

Всегда *Null* (*Cons* (x, y)) = **false**

Интерпретация схемы в случае списков

list<tel> p, k, n ; pp = **Nil**; kk = **Nil**;
g(x) = Tail (x); h(x, y)= ConsHd (x,y)= Cons (Head (x), y);
Null (x) ≡ (x=**Nil**)

```
// Схема программы:
p = pp; k = n;
while (k !=kk) {
   p = h (k, p);
   k = g(k);
}
```

```
// Программа:
//вход – список n, м.б. пустой
p = Nil; k = n;
while (k !=Nil) {  // !Null(k)
   p = ConsHd (k, p);
   k = Tail (k);
}
```

Результат: ? Reverse (n)

Ручная прокрутка программы при *n* = **(т о р г)**

```
// Программа:
//вход – список n, м.б. пустой
p = Nil; k = n;
while (k !=Nil) {  // !Null(k)
   p = ConsHd (k, p);
   k = Tail (k);
}
```

номер шага	*p*	*k*
0	**( )**	**(т о р г)**
1	**(т)**	**(о р г)**
2	**(о т)**	**(р г)**
3	**(р о т)**	**(г)**
4	**(г р о т)**	**( )**

Добавим утверждения, т.е. спецификацию (предусловие, постусловие) + инвариант цикла

```
// вычисление n! (n≥0)
p = 1; k = n;
// inv: fct(k) *p = fct(n)
// var: v (k)= k
while (k !=0) {
   p = k *p;
   k = k – 1;
} // p =fct(n)
```

fct(n)=n!

```
// Схема программы:
// вычисление f(n)
p = pp; k = n;
// inv: q (f(k),p) = f(n)
// var: v (k) ***
while (k !=kk) {
   p = h (k, p);
   k = g(k);
} // p = f(n)
```

Интерпретация схемы: *pp, kk* – константы.

$h(x, y) = x * y;$ $g(x) = x - 1;$ $q(x, y) = x * y;$

Примечание: почему $h(x, y)$ и $q(x, y)$ разные функции

Требования к инварианту и варианту цикла

Интерпретация схемы: pp, kk – константы.

$h(x, y) = x * y$; $g(x) = x - 1$; $q(x, y) = x * y$;

1) При входе $(p = pp)\ \&\ (k = n) \Rightarrow q\,(f(k), p) = f(n)$

Т.е. $q\,(f(n), pp) = f(n)$

2) При выходе $(q\,(f(k), p) = f(n))\ \&\ (k = kk) \Rightarrow (p = f(n))$

3) Свойство сохранения:

$\{(k\ != kk)\ \&\ (q\,(f(k), p) = f(n))\}$

$p = h\,(k, p)$; $k = g(k)$;

$\{(q\,(f(k), p) = f(n))\}$

Интерпретация схемы в случае списков

```
list<tel> p, k, n ; pp = Nil; kk = Nil;
g(x) = Tail (x);   h(x, y)= ConsHd (x,y)= Cons (Head (x), y);
q (x,y) = Concat(x,y)
```

```
// Схема программы:
// вычисление f(n)
p = pp; k = n;
// inv: q (f(k),p) = f(n)
// var: v (k)
while (k !=kk) {
   p = h (k, p);
   k = g(k);
} // p = f(n)
```

```
// Программа:  f(n)=Reverse(n)
//вход – список n, м.б. пустой
p = Nil; k = n;
// inv: q (f(k),p) = f(n)
// var: v (k)
while (k !=Nil) {  // !Null(k)
   p = ConsHd (k, p);
   k = Tail (k);
} // p = f(n)
```

Инвариант интерпретации со списками

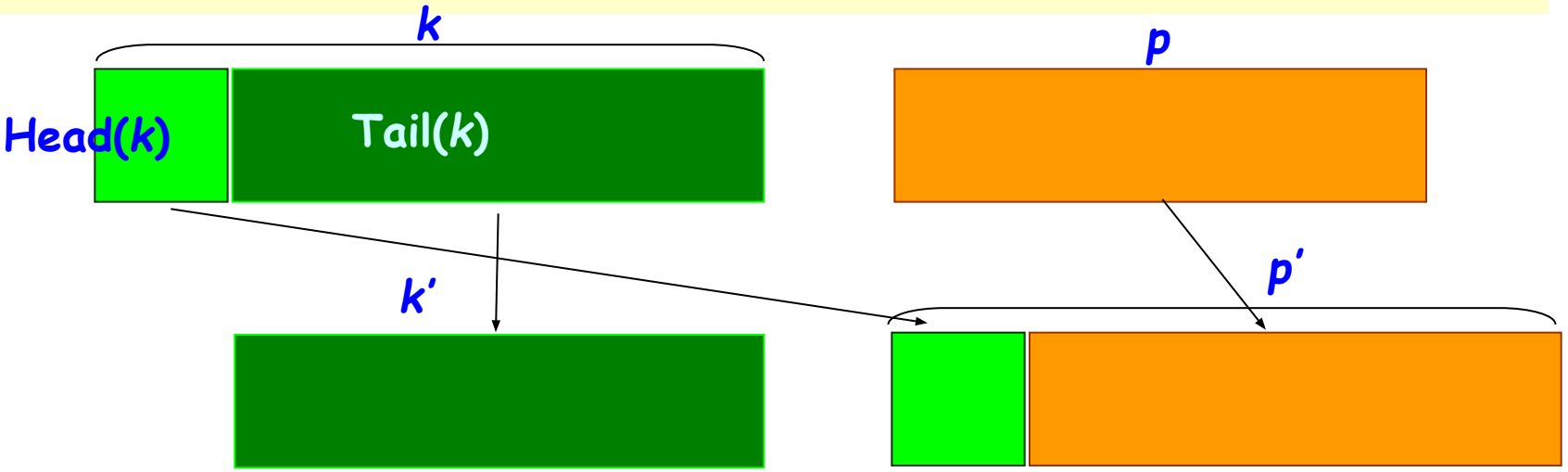
Concat (**(к а р)**, **(м у з)**) = **(к а р м у з)**,
Concat (**Nil**, *y*) =*y*, *Concat* (*x*,**Nil**) =*x*
q (*x*,*y*) = *Concat*(*x*,*y*)
inv: *q* (*f*(*k*),*p*) = *f*(*n*)
т.е. *Concat*(*Reverse*(*k*),*p*) = *Reverse*(*n*)

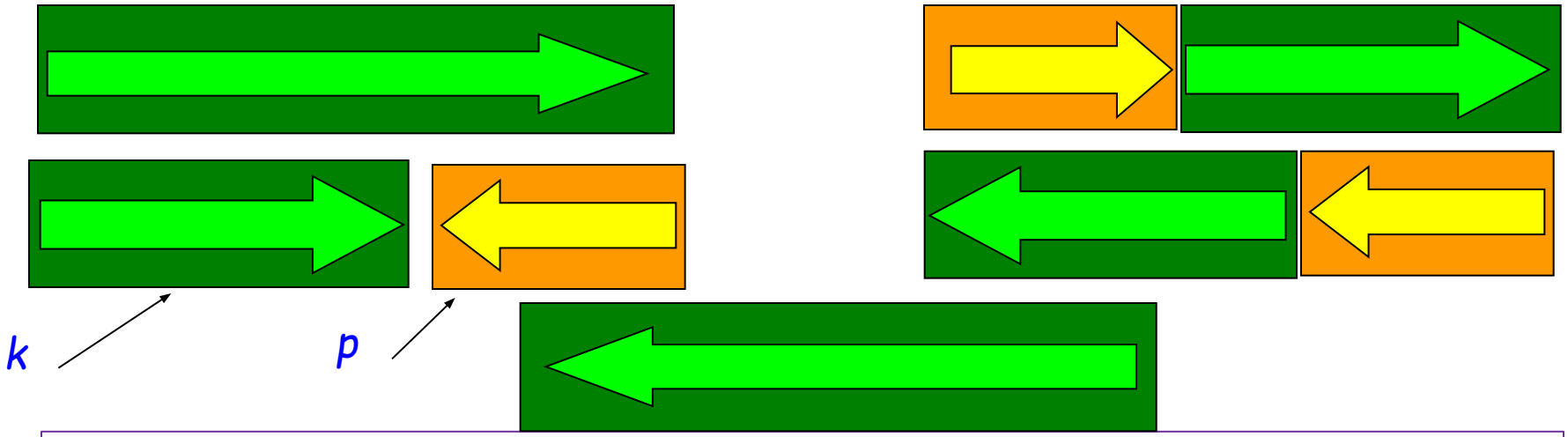
Если ввести инфиксное обозначение
s1 ⊕ *s2* = *Concat* (*s1*, *s2*),
то inv: *Concat*(*Reverse*(*k*),*p*) = *Reverse*(*n*) превратится в
Reverse(*k*) ⊕*p* = *Reverse*(*n*),
что легче интерпретировать как аналог
fct(*k*) **p* = *fct*(*n*).

Требования к варианту цикла

$v(k) \equiv length(k)$
?

Картинки к телу цикла и к инварианту





Reverse (n) = Concat (Reverse (k), p)