KYBERNETIKA ČÍSLO 1, ROČNÍK 4/1968

The Formulation of the Problem of Program Segmentation in the Terms of Pseudoboolean Programming

JAROSLAV KRÁL

The problem of (a priori) segmentation of a program is formulated. Two equivalent formulations are stated as programs with zero-one variables.

Let us have a program P for a digital computer. Let its length be N, i.e. the program is in the machine language expressed by N instructions. If N is large, a part of P must be placed on a backing store. We shall suppose that an information from a backing store can be called only in tracks of the length K and that the bounds of tracks on the backing store are given by the hardware of the computer. We shall assume that the backing store has practically unlimited capacity.

In the run time the segment with the executed instruction must be previously transferred into the main store. If the control of the computer after the execution of the instruction in one segment needs the instruction from another segment, an administrative program must be called verifying whether the new track is in the main memory. If it is not the case, the administrative program calls the new track from the backing store. The problem of segmentation is the collecting of all the instructions of the program into the tracks so as to minimize the number of calls of the administrative program. This problem can be stated mathematically as follows:

Find the covering of the set of instructions of the program by the sets $B_1, \ldots, B_s$ so that the number of elements in any B_i is not greater than K and that the number of cases, when – in the run time – an instructions in B_j must be executed after execution an instruction in B_i $(i \neq j)$, is minimal.

B_i is the collection of instruction in the i-th track (a track needs not contain just K instruction as some of its location can be "empty"). As for various executions of the program the number of jumps from an instruction I_i on to an instruction I_j varies, we must state the problem in terms of mean values. Let us assume that we know the values of S_{ij} where S_{ij} is the mean value of the number of cases that (in the run time) after the execution of the instruction I_i the instruction I_j is executed. For the problem of estimation of S_{ij} see [7]. The structure of the program is for our purposes completely described by the matrix (S_{ij}) or by an oriented graph $\mathscr{G}$ of N vertices $V_1, \ldots, V_N$ labelled by instructions $I_1, \ldots, I_N$. The edges (V_i, V_j) of $\mathscr{G}$ are labelled by S_{ij} which can be interpreted as flows through the edges. We shall assume that no instruction can be in two segments. This assumption is somewhat limiting but, on the other hand, avoiding this assumption would cause such unwanted consequencies as insertion of almost all procedures on the places of their activations, and so on. It is clear that for a given segmentation of the program P the mean number of cases when after execution of an instruction in one segment an instruction from another segment is executed (or more shortly the mean number of intersegment jumps) is given by the expression

(1) $$\sum_{s=1}^{Q} \sum_{i, I_i \in B_s} \sum_{j, I_j \notin B_s} S_{ij}$$

where B_s denotes the s-th track (or segment) and $\sum_{i, I_i \in B_s}$ denotes that the summation is taken over all i for which I_i belongs to B_s. The number Q of segments is not fixed. Now we can formulate the problem of segmentation in the following way:

Find the disjoint subsets $B_1, \ldots, B_Q$ of the set $\{1, 2, \ldots, N\}$,

$$\bigcup_{i=1}^{Q} B_i = \{1, 2, \ldots, N\}$$

(i.e. all B_i cover the set $\{1, 2, \ldots, N\}$) so that

(1) number of elements in each B_i is not greater than K.

(2) the value of

(2) $$\sum_{s=1}^{Q} \sum_{i \in B_s} \sum_{j \notin B_s} S_{ij}$$

is minimal.

We note that although Q is not given (fixed) we can take $Q \leqq [N/[K/2]]$ where [ ] denotes the integer part. This results from the fact that the mean number of instructions per one segment need not be less than $[K/2]$ (in the opposite case at least two segments can be joint into one) and that each instruction is just in one segment. As we can allow B_i to be empty, we can take $Q = [N/[K/2]] + 1$.

Theorem 1. *The problem of segmentation is equivalent to the following pseudoboolean program:*

(2a) $$x_i^j = 0, 1\,; \quad i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, Q\,,$$

(2b) $$\sum_{i=1}^{N} x_i^j \leqq K\,,$$

(2c) $$\sum_{j=1}^{Q} x_i^j = 1\,,$$

(2d) $$\sum_{i=1}^{N} \sum_{k=1}^{N} S_{ik} \sum_{j=1}^{Q} x_i^j x_k^j \to \max .$$

Proof. Associate with each B_j a set of zero-one variables

$$x_1^j, x_2^j, \ldots, x_N^j .$$

Let $x_i^j = 1$ if I_i belongs to B_j and $x_i^j = 0$ in the opposite case (i.e. the values of $x_1^j, \ldots$ $\ldots, x_N^j$) coincide with the values of the characteristic function of B_j). Then (2b) is equivalent to the requirement that B_j contains at most of K elements, (2c) holds if and only if all B_i, $i = 1, 2, \ldots, n$, are disjoint and cover the set $\{I_1, I_2, \ldots, I_N\}$ i.e. cover the whole program P. (1) is then equivalent to

(3) $$\sum_{j=1}^{Q} \sum_{i=1}^{N} \sum_{k=1}^{N} S_{ik} x_i^j (1 - x_k^j)$$

as $x_i^j(1 - x_k^j)$ is equal to 1 if and only if $I_i \in B_j$ and $I_k \notin B_j$. But (3) is minimal if and only if the expression

(4) $$\sum_{i=1}^{N} \sum_{k=1}^{N} S_{ik} \sum_{j=1}^{Q} x_i^j x_k^j$$

is maximal. QED.

We find now the formulation of the problem of segmentation as a linear program in zero-one variables. We shall prove.

Theorem 2. *The problem of segmentation is solved if the solution of the following linear program in zero-one variables is known*:

(5a) $$x_{ij} = 0, 1 \quad \textit{for} \quad i, j \ = 1, 2, \ldots, N ,$$

(5b) $$x_{ij} = x_{ji} \quad \textit{for} \quad i, j \ = 1, 2, \ldots, N ,$$

(5c) $$x_{ii} = 0 \quad \textit{for} \quad i \ = 1, 2, \ldots, N ,$$

(5d) $$x_{ik} - x_{jk} \leqq x_{ij} \quad \textit{for} \quad i, j, k = 1, 2, \ldots, N ,$$

(5e) $$\sum_{j=1}^{N} x_{ij} \geqq N - K , \qquad i = 1, 2, \ldots, N ,$$

(5f) $$\sum_{i=1}^{N} \sum_{j=1}^{N} S_{ij} x_{ij} \to \min .$$

Proof. Let us put in (3)

$$\sum_{s=1}^{Q} x_i^s (1 - x_j^s) = x_{ij} ;$$

then (2c) implies

$$\sum_{s=1}^{Q} x_i^s (1 - x_j^s) = x_i^{s_i} (1 - x_i^{s_i}) = x_{ij} = 0, 1$$

where s_i denotes the unique segment containing I_i. $x_{ij} = x_{ji} = 0$ if I_i and I_j belong to the same segment and $x_{ij} = x_{ji} = 1$ if I_i and I_j belong to different segments. (2b) is then equivalent to the system of inequalities (5e), because

$$(6) \qquad \sum_{j=1}^{N} x_{ij} = \sum_{j=1}^{N} x_i^{s_i}(1 - x_i^{s_i}) = N - \sum_{j=1}^{N} x_i^{s_i} .$$

Further (3) is equivalent to (5f). It only remains to express the conditions (2c). (2c) is equivalent to the requirement that each instruction of the program belongs to one and only one segment. But then for a fixed i

$$x_{ij} = 1 - \chi_{s_i}(j) ;$$

$\chi_{s_i}(j)$ is the characteristic function of the segment containing I_i. If I_i and I_j belong to the same segment (we know that this is equivalent to requirement $x_{ij} = 0$) we have $\chi_{s_i} = \chi_{s_j}$ and hence $x_{ik} = x_{jk}$; $k = 1, 2, \ldots, N$. So for Q disjoint segments the matrix $(x_{ij})_{i,j=1}^{N}$ contains just Q mutually different rows and if $x_{ij} = 0$ then i-th and j-th row coincide.

Now we prove

Lemma 1. *Let a matrix of zero-one variables* $(x_{ij})_{i,j=1}^{N}$ *fulfil the conditions* (5b), (5c) *and if* $x_{ij} = 0$ *then* $x_{ik} = x_{jk}$,$k = 1, 2, \ldots, N$. *Let* $X_{i_1}, \ldots, X_{i_Q}$ *be all mutually different rows of* (x_{ij}). *Let* $\chi_k(j) = 1 - x_{i_k j}$ *and let* B_k *be a subset of* $\{1, 2, \ldots, N\}$ *with the characteristic function* χ_k. *Then the sets* B_k, $k = 1, 2, \ldots, Q$, *form a disjoint covering of* $\{1, 2, \ldots, N\}$

Proof. $\{B_k \mid k = 1, 2, \ldots, Q\}$ forms a covering as $x_{ii} = 0$ so each i belongs to at least one B_k. Now, if there exist s and m so that $s \neq m$, $B_s \neq B_m$ and $B_m \cap B_s \neq \emptyset$ (i.e. B_k, $k = 1, 2, \ldots, Q$, are not mutually disjoint) then there exist j, r, q so that $j \in B_s \cap B_m$, $r \in B_s$, $q \in B_m$ and $r \notin B_m$, $q \notin B_s$. Then $x_{rj} = x_{jq} = 0$, $x_{rq} = 1$ but from the assumption of the lemma follows that

$$x_{rk} = x_{jk} \quad \text{for} \quad k = 1, 2, \ldots, N ,$$
$$x_{jk} = x_{qk} \quad \text{for} \quad k = 1, 2, \ldots, N$$

and thus

$$x_{rk} = x_{qk} \quad \text{for} \quad k = 1, 2, \ldots, N .$$

For $k = r$

$$x_{rr} = x_{qr} = x_{rq} = 0$$

what is a contradiction.

We now return to the proof of theorem 2. From lemma 1 and from the first half of the proof of theorem 2 it follows that $\{B_k,\ k = 1, 2, \ldots, Q\}$ forms the disjoint covering if the matrix (x_{ij}) fulfils the assumptions of the lemma 1. So these assumptions imply (2c). As $x_{ij} = x_{ji}$, the inequalities (5d) imply that from $x_{ij} = 0$ the equality $x_{ik} = x_{jk}$ follows for $k = 1, 2, \ldots, N$ and the theorem is proved.

In the conclusion of the paper a few remarks will be convenient. The numbers S_{ij} must fulfil some limiting requirements (see [1] or [5]). For example, for the majority of $j \in \{1, 2, \ldots, N\}$ the equality

$$\sum_{i=1}^{N} S_{ij} = \sum_{i=1}^{N} S_{ji} \tag{7}$$

must hold. (7) holds for all I_j in which the work of the program can neither start nor end. For a small number of j either

$$\sum_{i=1}^{N} S_{ij} = \sum_{i=1}^{N} S_{ji} + p_j\,, \quad p_j > 0 \tag{8}$$

holds if the work of the program ends with probability p_j in instruction I_j, or

$$\sum_{i=1}^{N} S_{ij} + q_j = \sum_{i=1}^{N} S_{ji}\,, \quad q_j > 0 \tag{9}$$

holds if the work of the program starts with probability q_j in the instruction I_j. The program can be therefore described by the oriented graph with flows S_{ij}. The total flow through this graph is one because $\sum p_i = \sum q_i = 1$. In many cases there exist one and only one vertex V_s which fulfils (8) and one vertex V_E fulfilling (9). Without loss of generality we can assume that V_s is V_1 and V_E is V_N. Then

$$\sum_{i=1}^{N} S_{ij} = \sum_{i=1}^{N} S_{ji} \quad \text{for} \quad j = 2, 3, \ldots, N\,,$$

$$1 + \sum_{i=1}^{N} S_{i1} = \sum_{i=1}^{N} S_{1i}\,,$$

$$\sum_{i=1}^{N} S_{iN} = \sum_{i=1}^{N} S_{Ni} + 1\,,$$

For this case an effective algorithm of segmentation for segments B_i of the form

$$B_i = \{I_k \mid i_d \leqq k < i_u\}$$

was developed in [1].

(Received January 16th, 1967.)

REFERENCES

[1] J. Král: To the Problem of Segmentation of Program. In Information Processing Machines. Publishing House of the Czechoslovak Academy of Sciences, Prague 1965, 140—149.

[2] Claude Berge: Theorie des Graphes et ses Applications, Dunod, Paris 1958.

[3] Peter L. Ivănescu: Some Network Flow Problems Solved with Pseudoboolean Programming. Operations Res. *13* (1965), 3.

[4] Egon Balas: An Additive Algorithm for Solving Linear Programs with Zero-One Variables. Operations Res. *13* (July-Aug. 1965), 517—549.

[5] J. Král: One Method for Estimation of Frequencies of Jumps in a Program. (To appear.)

Formulace problému segmentace programu jako problému pseudobooleovského programování

JAROSLAV KRÁL

Předpokládejme, že program *P* pro číslicový počítač *C* je v jazyce stroje *C* vyjádřen *N* instrukcemi. Je-li *N* velké, nemůže být program *P* v době výpočtu umístěn celý v operační paměti počítače *C* a musí být rozdělen na části – segmenty tak, že v každém okamžiku výpočtu je v operační paměti počítače C jen několik málo segmentů. Jestliže je v době výpočtu provedeno předání řízení s instrukce v jednom segmentu na instrukci v druhém segmentu, musí být volán administrativní program, který prověří, zda je „nový" segment již v operační paměti a je-li třeba, přepíše tento segment z vedlejší paměti do operační paměti a provede předání řízení. Problémem je rozdělit program do segmentů (čili segmentovat program) tak, aby byl střední počet volání administrativního programu během výpočtu podle *P* minimální. V článku je nalezena matematická formulace tohoto problému a je ukázáno, že problém segmentace programu může být pro disjunktní segmenty formulován jako úloha nalezení extrému kvadratické nebo lineární formy v proměnných nabývajících hodnot 0 nebo 1. Problém segmentace programu může být formulován jako úloha pseudobooleovského programování (viz [33] a [4]). Optimální segmentace programu může být tedy nalezena metodami pseudobooleovského programování.

Jaroslav Král, prom. matematik, Ústav výpočtové techniky ČSAV—ČVUT, Horská 3, Praha 2.