ZKUŠENOSTI Z TVORBY A LADĚNÍ PROGRAMŮ VTV POD OPERAČNÍM SYSTÉMEM 4. GENERACE

Ing. Jan Kučera, Ústav fyzikální metalurgie ČSAV, Brno

1. Úvod

Výpočtové oddělení Ústavu fyzikální metalurgie ČSAV má přibližně patnáctileté zkušenosti s tvorbou matematického programového vybavení. Zabývá se zejména implementací vlastních i převzatých rozsáhlých programových systémů - viz /4/. Během uvedené doby využívalo vlastní počítač 2. generace, externí počítače řady JSEP a v současné době pracuje na velmi moderním počítači ICL 2950/10. Cílem tohoto příspěvku je poukázat na změny v charakteru programátorské práce, které s sebou přinesl přechod na využívání počítače s velmi pokročilým operačním systémem umožňujícím položit těžiště programátorské práce na interaktivní využívání počítače.

2. Zkušenosti z práce pod OS/EC

Přechod s počítače 2. generace na počítače JSEP vybavené operačním systémem OS přinesl do programátorské práce řadu progresivních prvků, jako možnost využívání jazyka Fortran ve verzi značně obohacené vůči normě, využívání systému knihoven zdrojových programů či možnost vytvářet JCL procedury řídící ladění i běh hotových úloh. Přesto při práci pod standardní verzí OS znající pouze režim dávkového zpracování se projevovala řada faktorů značně zpomalujících programátorskou práci.

Nejvážnější brzdou se jeví pomalá obrátka úloh. I u organizace s vlastním počítačem lze ztěží předpokládat více než 1 až 2 obrátky laděného programu denně, na externím počítači 1 až 2 obrátky týdně. Dalším velmi nepříjemným faktem je diagnostika chyb zjištěných během výpočtu. Analýza hexadecimálních výpisů obsahu paměti, což je často jediný prostředek k odhalení chyby, je časově náročná a vyžaduje znalosti, které průměrný programá-

tor nemívá.

Vezmeme-li v úvahu další faktory, jako nedokonalost textového editoru (utility IEBUPDTE), zaplňování knihoven starými versemi programů a z toho plynoucí nutnost časté reorganizace knihoven a příliš komplikovaný styk programátorů s operačním systémem, dojdeme k závěru, že tento systém je velmi vzdálen od ideálu "uživateli přátelského" systému.

3. Práce pod operačním systémem 4. generace

Na počítačích ICL řady 2900 jsou k dispozici velmi pokročilé operační systémy řady VME (Virtual Machine Environment), mající mnoho společného s moderními operačními systémy dalších výrobců. Na rozdíl od systémů založených na systémech IBM (DOS/OS) se jedná o systémy, které jsou od základu koncipovány s vědomím, že jsou určeny pro počítače s virtuální pamětí, interaktivními terminály a pro uživatele programující ve vyšších programovacích jazycích (assembler není ani dodáván). Zájemci o podrobnější seznámení se systémy VME je naleznou v lit. /1/ a /2/; zde se zmíníme pouze o dílčích složkách systému, které podle autorova názoru nejpronikavěji ovlivnily charakter programátorské práce. Výklad budeme ilustrovat na ladění následujícího programu pro nalezení největšího společného dělitele 2 čísel:

```
      INTEGER GCD
      READ(5,90)I,J
   90 FORMAT(2I5)
      K=GCD(I,J)
      WRITE(6,91)I,J,K
   91 FORMAT(4HGCD(,I5,1H,I5,2H)
      STOP
      END
```

```
      INTEGER FUNCTION GCD(I,J)
      II=I
      JJ=J
   10 GCD=MOD(II,JJ)
      IF(GCD.LT.0)RETURN
      II=JJ
      JJ=GCD
      GOTO 10
      END
```

(Poznamenejme, že v příkaze IF podprogramu GCD je chyba: místo .LT. má být správně .EQ.; tato chyba má za následek dělení nulou v příkaze s návěštím 10.)

3.1. Hlášení chyb

Chyby, k nimž dochází během výpočtu, jsou hlášeny zásadně tak, aby text byl srozumitelný bez znalosti strojového jazyka. Uživatel má možnost volit stupeň detailnosti chybových zpráv.

Hlášení chyby při výpočtu výše uvedeného programu může mít (po vypuštění několika nepodstatných drobností) např. tvar:

```
DIAGNOSTIC REPORT 1 ON 1983/03/25 AT 09.12.43
INTERRUPT ERROR: -500
DESCRIPTION: ZERO DIVIDE

PROGRAM AT LINE: 12
IN PROCEDURE:GCD COMPILED ON 1983/03/25,AT 09.11.12

REPORT OF CURRENT STATE OF PROGRAM

FORTRAN FUNCTION GCD AT LINE 12
     ( MODULE GCD COMPILED ON 1983/03/25 AT 09.11.12 )
I    = 34          II  = 17          J    = 85          JJ   = 0

FORTRAN PROGRAM ICL9HPMAIN AT LINE  4
     ( MODULE ICL9HPMAIN COMPILED ON 1983/03/25 AT 09.11.12
I    = 34          J   = 85          K    = 0

PROGRAM TERMINATED
```

Uživatel má možnost kromě pořízení uvedeného výpisu (nebo místo něj) ošetřit chybu vlastním podprogramem nebo ji nechat ignorovat.

Uvedený systém hlášení chyb je o to cennější, že hardware hlídá i chyby, které u jiných počítačů (např. JSEP) hlídány nejsou - například přetečení indexu nad horní mez pole nebo pokus o zápis do instrukční části programu.

3.2. Interaktivní testovací systém ITS

Tento programový produkt - podrobnosti viz /5/ - umožňuje interaktivně zasahovat do průběhu programu psaného ve vyšším programovacím jazyce. Dává například možnost zastavovat program na zvolených řádcích nebo při změně hodnoty zvolených proměnných, v místě zastavení vypisovat a měnit hodnoty proměnných, spouštět části programu ve zvoleném pořadí apod. Komunikace s ITS je opět v termínech zdrojového jazyka. Pokud by například méně zkušený programátor neodhalil zdroj chyby uvedené v předchozím na základě chybové zprávy, má možnost prostřednictvím ITS vést následující dialog (řádky předznačené I jsou zadávány programátorem v průběhu výpočtu pod ITS):

```
PREPARING I.T.S. FOR MODULE ICL9HFMAIN COMPILED ON 1983/03/25 AT
                                                    09.14.09
AT START OF MODULE ICL9HFMAIN - STATE REQUIREMENTS
I DEFAULT GCD              (následující se netýká ICL9HFMAIN, ale GCD)
PREPARING I.T.S. FOR MODULE GCD COMPILED ON 1983/03/25 AT 09.14.09
I MONITOR GCD              (zastav při každé změně hodnoty GCD)
I CONTINUE                 (pokračuj ve výpočtu)
MONITOR HALT AT LINE  13 - CHANGED GCD
I PR II,JJ,GCD             (tiskni hodnoty zvolených proměnných)
 34 85 34
I CONTINUE
MONITOR HALT AT LINE  13 - CHANGED GCD
I PR II,JJ,GCD
 85 34 17
I CONTINUE
MONITOR HALT AT LINE  13 - CHANGED GCD
I PR GCD
 0
I CONTINUE
(Nyní se opět vytiskne chybové hlášení o dělení nulou jako
 v bodě 3.1. a výpočet pokračuje. Tím je zřejmé, že se příkaz
 IF neprovedl správně - pro GCD = 0 je nutno z podprogramu
 vystoupit. Proto výpočet ukončíme:)
MONITOR HALT AT LINE  13 - CHANGED GCD
I TERMINATE
```

3.3. Interaktivní textový editor

Pro interaktivní ladění programů je nutný kvalitní textový editor. Editor používaný v operačních systémech VME je špičkové úrovně. Začátečník se může spokojit s postupným zobrazováním textu na terminálu, jeho opravami na obrazovce hardwarovými prostředky příslušného terminálu a odesílánim opraveného textu. Naproti tomu zkušený programátor ocení i mohutný editovací jazyk, který staví tento editor téměř na úroveň makroprocesoru. Následující editovací příkazy například změní všechny příkazy PRINT na příkazy WRITE(6,... a do všech příkazů READ(... doplní klausuli END=999:

```
-E((                       až do konce programu prováděj následující:
C?PRINT?,                  obsahuje-li řádek text 'PRINT'
R?PRINT?WRITE(6,?,         nahraď text 'PRINT' textem 'WRITE(6,'
R?,?)?                     nejblíže následující čárku nahraď závorkou
;C?READ(?,                 jinak obsahuje-li řádek text 'READ('
T.?)?,                     zkopíruj jej až k první pravé závorce
I?,END=999?,               vlož na toto místo text',END=999'
T+1                        a zbytek řádku zkopíruj beze změny
;T+1))                     jinak (řádek bez PRINT a READ) zkopíruj
                           řádek beze změny
E                          ukonči editování
```

Uvedený příklad nezpracuje správně příkazy PRINT bez seznamu proměnných, lze však samozřejmě napsat editovací příkazy tak, aby i tato varianta byla opravena správně.

3.4. Jazyk pro řízení úloh

V jazyce pro řízení úloh (JCL) se snad nejvíce projevuje rozdíl oproti operačním systémům používaným na počítačích JSEP. Tyto jazyky (KCL či SCL podle typu systému) jsou skutečnými programovacími jazyky vyšší úrovně včetně vlastních proměnných (přístupných např. z programů psaných v Cobolu), blokové struktury, příkazů skoku (IF, GOTO), cyklů a podprogramů či maker. Program v jazyce SCL lze dokonce zkompilovat do běžného tvaru přeloženého (object) modulu, který může být vyvolán z programů psaných v běžných programovacích jazycích.

Kromě těchto obecných možností obsahují samozřejmě jazyky KCL/SCL prostředky specificky určené k tomu, čemu tyto jazyky slouží - totiž k řízení úloh. Stojí za zmínku, že veškerá komunikace se systémem se děje v jazyce KCL/SCL, bez ohledu na to, zda jde o komunikaci programátora, uživatele, operátora nebo systémového programátora. Samozřejmě, že ne každý má však pravomoc vydávat jakýkoli příkaz.

3.5. Vyšší programovací jazyky

Již dříve jsme se zmínili o tom, že uživatelé nemají k dispozici jazyk typu assembler. O to větší pozornost je věnována vyšším programovacím jazykům. Z běžných jazyků je k dispozici Fortran, Cobol, Algol 60, Algol 68, Basic a Pascal. Implementace Fortranu je asi uprostřed mezi "běžným" Fortranem (podle normy z roku 1966) a Fortranem 77 (k dispozici je rovněž úplný Fortran 77). Velmi cenným rozšířením ICL-Fortranu proti normě 1966 je zavedení datového typu CHARACTER. Navíc je návaznost na operační systém vyřešena takovým způsobem, že v řadě případů lze program převést z jednoduché do dvojnásobné nebo čtyřnásobné přesnosti pouhým zadáním vhodného režimu překladu bez zásahu do zdrojového fortranského textu.

3.6. Nejen klady, ale i problémy

Nic na světě není ideální - ani operační systémy VME. Hlavní zdroj problémů spočívá v malé kompatibilitě s běžnějšími počítači.

Zejména při přebírání programů původně psaných pro IBM 360/370 či JSEP se často naráží na odchylky od normy Fortranu, které na JSEP jsou přípustné, zatímco na ICL nikoli. Klasickým příkladem je zlozvyk deklarovat jednorozměrná pole v podprogramu jako pole o jednom prvku, např. REAL X(1). V důsledku hardwarových kontrol zmíněných v bodě 3.1 jsou podobné konstrukce zdroji hlášení chyb při výpočtu. I když tyto kontroly lze vhodným režimem překladu potlačit, znamenají u větších programových systémů zdržení v etapě oživování a nutnost vést evidenci o tom, jakým režimem je nutno který podprogram překládat.

4. Závěr - a co JSEP?

Mnohý z účastníků semináře Programování 83 si asi během čtení předchozích kapitol řekl něco jako: "To je moc hezké, ale jak se to týká nás, co nepracujeme na popisovaném systému?" Autor by proto chtěl upozornit na to, že i na běžně dostupných počítačích lze zajistit možnost pracovat způsobem více či méně podobným tomu, o němž bylo v příspěvku referováno.

V síti počítačů ČSAV založené v současné době zejména na počítačích EC 1040 se běžně používá terminálů k interaktivnímu ladění programů a zadávání úloh pro dávkové zpracování. Jedná se zejména o inteligentní terminály na bázi minipočítače JPR 12 vyvinuté Střediskem výpočetní techniky ČSAV. Terminály jsou obsluhovány řídicím programem umožňujícím práci se soubory včetně editování, překladů Fortranem, PL1, Assemblerem a Algolem W a v případě Algolu W i menších výpočtů. Kromě toho umožňují zadávat úlohy do dávkového zpracování pomocí subsystému HASP. Na počítači Astronomického ústavu ČSAV v Ondřejově jsou používány konversační terminály řady MERA 7900 pracující pod Time Sharing Option (TSO) operačního systému OS/IBM. Kromě toho se rozvíjí i napojování satelitních počítačů na centrální počítač SVT ČSAV.

V roce 1982 vyšla velmi zajímavá kniha /3/ popisující systém interaktivní práce ve Fortranu pod operačním systémem OS. Systém zahrnuje komponenty podobné těm, o nichž bylo referováno v 3. kapitole tohoto příspěvku včetně 2 Fortranských kompilátorů vhodných

pro interaktivní práci, textového editoru a dialogového ladicího systému.

Lze proto doufat, že i na počítačích u nás běžně dostupných se postupně bude prosazovat interaktivní práce při ladění programů, která ladění mnohonásobně zrychluje a zvyšuje tak podstatně efektivitu programátorské práce.

5. Literatura

/1/ Z.Rusín: Výpočetní systémy příštích let a jejich dopad na profesní sféru. Programování '82, Dům techniky ČSVTS, Ostrava 1982.

/2/ J.Kučera, P.Setínek: Počítače 4. generace ICL a jejich operační systémy. Zborník prednášok 6. sympózia Algoritmy 81. JSMF pri SAV, Bratislava, 1981.

/3/ Z.S.Brič, D.V.Kapilevič, O.G.Těrechova: Programmirovanije na Fortrane ES EVM v režime razdělenija vremeni. Finansy i statistika, Moskva, 1982.

/4/ J.Hřebíček, I.Kopeček: Tvorba, ladění a testování numerického software, Programování '83, Dům techniky ČSVTS, Ostrava, 1983.

/5/ I.Kopeček: Verifikace programů metodou interaktivního testování v systému ICL 2950/10. MOP 82 - díl 4. Dům techniky ČSVTS, Žilina, 1982.