Programok párhuzamos feldolgozása

Antal Iványi
Eötvös Loránd University
Department of Computer Algebra
H-1117, Budapest, Hungary
Pázmány sétány 1/C
email: tony@compalg.inf.elte.hu

(2010. október 17.)

Az alábbiakban összefoglaljuk a programok párhuzamos feldolgozásával kapcsolatban eddig publikált eredményeinket, és néhány megoldandó feladatot.

Az átfedéses memória lényege, hogy egyidejű működésre képes blokkokból áll (ezeket az $1, 2, \ldots, n$ számokkal modellezzük) [8, 21].

Vizsgálataink fő célja az, hogy megadjuk a számítógép sebességét, mint a hardver paraméterek (blokkok száma, blokkok rekeszeihez és a blokkok speciális regisztereihez való hozzáférés ideje, műveletvégzési idők), a szoftver paraméterek (programok viselkedésének jellemző adatai [8]) és a feldolgozó algoritmusok függvényét.

Az eredményeket első sorban aszerint csoportosítjuk, hogy a párhuzamosan futó programokat (hivatkozási sorozatokat) prioritásos vagy prioritás nélküli algoritmussal dolgozzuk-e fel.

Paraméter lehet a blokkok száma, illetve az egyes blokkokhoz tartozó speciális regiszterek száma is.

Az alábbiakban összefoglaljuk a programok párhuzamos feldolgozásával kapcsolatban eddig publikált eredményeinket, és néhány megoldandó feladatot.

Az átfedéses memória lényege, hogy egyidejű működésre képes blokkokból áll (ezeket az $1, 2, \ldots, n$ számokkal modellezzük) [8, 21].

Vizsgálataink fő célja az, hogy megadjuk a számítógép sebességét, mint a hardver paraméterek (blokkok száma, blokkok rekeszeihez és a blokkok speciális regisztereihez való hozzáférés ideje, műveletvégzési idők), a szoftver paraméterek (programok viselkedésének jellemző adatai [8]) és a feldolgozó algoritmusok függvényét.

Az eredményeket első sorban aszerint csoportosítjuk, hogy a párhuzamosan futó programokat (hivatkozási sorozatokat) prioritásos vagy prioritás nélküli algoritmussal dolgozzuk-e fel.

Paraméter lehet a blokkok száma, illetve az egyes blokkokhoz tartozó speciális regiszterek száma is.

1 Prioritásos feldolgozás átfedéses memóriával

Alapesetben azt tételezzük fel, hogy a processzor minden időegységben a hivatkozási sorozat maximális prefixét dolgozza fel annak figyelembe vételével, hogy minden lépésenként minden blokkra legfeljebb egyszer lehet hivatkozni.

1.1 Egy sorozat, n blokk, RAN programviselkedés

Tegyük fel, hogy a programok minden utasításához egyforma valószínűséggel kell a blokkokhoz fordulni.

Lemma 1 [7, 9, 10, 12, 16, 23] *Ha $n \geq 1$, akkor a $V(n)$ sebesség*

$$V(n) = \sum_{i=1}^{n} p_i(n) i, \tag{1}$$

ahol

$$p_i(n) = \frac{n! i}{(n-i)! n^{i+1})} \ . \tag{2}$$

Tétel 2 [7] *Ha $n \geq 1$, akkor*

$$\left| V(n) - \sqrt{\frac{\pi n}{2}} \right| < 1. \tag{3}$$

Tétel 3 [1, 15, 16, 14, 18] *Ha $n \geq 1$, akkor*

$$V(n) = \sqrt{\frac{\pi n}{2}} - \frac{1}{3} + \epsilon(n), \tag{4}$$

ahol $\epsilon(n)$ n növekedtével monoton csökkenve tart a nullához.

1.2 Egy sorozat, n blokk, STEP programviselkedés

Tegyük fel, hogy az i-edik blokk után p valószínűséggel következik az $(i+1)$-edik blokk $(\mod n)$, és a többi átmenet valószínűsége egyforma [2].

1.3 Két sorozat, n blokk

Ha két sorozatot dolgozunk fel prioritással, akkor legyen az első sorozat feldolgozási sebessége $V_1(n)$, a második sorozaté pedig $V_2(n)$.

Tétel 4 [9] *Ha* $n \geq 1$, *akkor*

$$V_1(n) = V(n) \quad \textit{és} \quad V_2(n) = \frac{V(n)}{2} + O(1). \tag{5}$$

1.4 Több bináris program, egyenletes eloszlás

[19]

1.5 Több bináris program, nem egyenletes eloszlás

[20, 21]

1.6 Több blokk, több program

Feladat: határozzuk meg az egyes programok $V_i(n,k)$ sebességét $n \geq 3$ blokk és $k \geq 3$ program esetén [22].

2 Prioritás nélküli feldolgozás átfedéses memóriával

Tegyük fel, hogy a processzor minden időegységben minden blokkhoz legfeljebb egyszer fordulhat, és a programok $(i+1)$-edik hivatkozását egy adott lépésben csak akkor dolgozhatjuk fel, ha a megelőző hivatkozásokat legkésőbb az adott lépésben már feldolgoztuk.

2.1 Átlagsebesség két bináris sorozat esetén

Tegyük fel, hogy két n hosszúságú bináris sorozatot kell feldolgozni prioritás nélkül. Legyen $V(n)$ az n hosszúságú bemenet esetén az átlagos sebesség optimális kiszolgálás esetén.

Ekkor ismert:

a) a $V(n)$ sebesség szigorúan monoton nő [13];

b) konkrét n hosszúságú bemenet esetén az optimális kiszolgálás polinomiális idő alatt meghatározható [13];

c) az átlagsebesség különböző közelítő feldolgozási algoritmusok (mint SIM_k, MEM_k és ALT_k) esetén [13].

d) [4] ha n hosszú sorozatok esetén minden bemenetet annyi időegység alatt dolgoznánk fel, mint a nullák és egyesek számának maximuma, akkor egy elem feldolgozásának átlagos időigénye

$$T_{opt} = n2^{2n} + (n+1)\binom{n+1}{n-1} \tag{6}$$

lenne, ahonnan

$$T_{opt} \geq \frac{1}{2} + \frac{c_n}{\sqrt{n}} \tag{7}$$

következik, ahol $c_n \sim 0.3$

Kérdés:

Hová tart $V(n)$, ha n minden határon túl nő.

2.2 Két számítógép sebességének aránya blokkonként több regiszter esetén

[3],

3 Párhuzamos folyamatok ütemezése

[6]

References

[1] T. T. Cirulis, A. Iványi On the monotonity of a "small" function. In: *Fourth Conference of Program Designers* (ed. by A. Iványi, Budapest, June 1–3, 2010). ELTE TTK, Budapest, 2010, pp. 171–180. ⇒2

[2] G. P. Egorychev, A. M. Iványi, A. I. Makosii, Analysis of two combinatorial sums that characterize the speed of a computer with block memory. (Russian) Ann. Univ. Sci. Budapest. Sect. Comput. 7 (1987), 19–32. MR0975341 (90d:68062), 68R05 (05A19 68N25); Zbl 0712.05008, 05A19 68R05 68N25. ⇒2

[3] A. Iványi, On dumpling-eating giants. *Finite and infinite sets,* Vol. I, II (Eger, 1981), 379–390. Colloq. Math. Soc. János Bolyai, 37, North-Holland, Amsterdam, 1984. MR0818249 (87f:68004) ⇒4

[4] A. Iványi, On the minimization of the overhead cost at the parallel execution of programs. *Programmirovanie* **10** (1) (1984) 63–68. ⇒4

[5] A. Iványi, Modeling of program runs using popular Markov chains. (Russian). *Vestnik Moskov. Univ. Ser. XV Vychisl. Mat. Kibernet.* 1984, (1) 59–65. MR0739324 (85j:68007) 68M20 (60J99 68Q10), Zbl 0529.68077, MSC2000: 68U20 68N99. A. Iványi, Simulation of program execution using popular Markov chains. (English) Mosc. Univ. Comput. Math. Cybern. 1984, No.1, 77-85 (1984). Zbl 0555.68070, MSC2000: 68U20 68N99. ⇒

[6] A. Iványi, Density of safe matrices. *Acta Univ. Sapientiae Math.* **1** (2) (2009) 121–142. MR2521183 (2010h:60266), 60K35 (68M20 82B43), Zbl pre05606303, 60G50 82B43 68M20 ⇒4

[7] A. Iványi, I. Kátai, Estimates for speed of computers with interleaved memory systems, *Annales Univ. Sci. Budapest., Sectio Mathematica*, **19** (1976), 159–164. MR0520987 (58 #25112), 68A20 (10K05). ⇒2

[8] A. Iványi, I. Kátai, Átfedéses memóriájú számítógépek tejesítményéről. *Alkalmaz. Mat. Lapok* **3** (1) (1977) 1–11. MR0502386 (58 #19454) 68A99. ⇒1

[9] A. Iványi, I. Kátai, Processing of random sequences with priority. *Acta Cybernet.* **4,** (1) (1978/79) 85–101. MR0521454 (80d:68029), 68B20 (60J99).
http://www.inf.u-szeged.hu/actacybernetica/prevvols/4_1/4_1.pdf ⇒2, 3

[10] A. Iványi, I. Kátai, On the performance of computers with interleaved memory. *Selected Papers on Operating Systems: Theory and Practice* (Lectures, Visegrád Winter School, Visegrád, 1978), pp. 205–216, Számító. Kutató Int., Budapest, 1978. MR0540592 (80j:68019) 68B20 (68J10) ⇒ 2

[11] A. Iványi, I. Kátai, I. Parallel processing of random sequences with priority. The First Pannonian Symposium on Mathematical Statistics (Bad Tatzmannsdorf, 1979), pp. 122–139, *Lecture Notes in Statist.*, **8**, Springer, New York-Berlin, 1981. MR0621147 (83c:60017) 60C05 (60F05). ⇒

[12] A. Iványi, I. Kátai, Processing of independent Markov-chains. *Ann. Univ. Sci. Budapest. Sect. Comput.* **3** (1982), 33–46. MR0749267 (85k:60090) 60J10 (60J20 68M20 90C40); Zbl 0539.60092, 60K20 60J10 60K25. ⇒2

[13] A. Iványi, I. Kátai, Modeling of prioritiless processing in an interleaved memory with a perfectly informed processor. *Autom. Remote Control*, (4) (1985) 520-526. Zbl 0581.68035: 68M20 . Az *Avtomatika i Telemehanika* című orosz nyelvű folyóirat 1985. évi 4. száma 129–136. oldalainak fordítása.
http://compalg.inf.elte.hu/ tony/Kutatas/InterleavedMemory/Interleaved-IK-84.pdf ⇒3

[14] A. Iványi, I. Kátai, Quick testing of random variables. In: *Proceedings of ICAI'2010* (Eger, January 27–30). To appear. ⇒2

[15] A. Iványi, I. **and** Kátai, Testing of uniformly distributed vectors. In: *Abstracts of János* Bolyai Memorial Conference (Budapest, August 28-30, 2010), 47–47. ⇒2

[16] A. Iványi, I. Kátai, Testing of random matrices, *Annales Univ. Sci. Budapest., Sectio Computatorica* (submitted).
http://compalg.inf.elte.hu/ tony/Kutatas/TAMOP/Papers-in-journals/.
⇒2

[17] A. Iványi, Z. László, On the power of an electronic computer with interleaved memory. (Russian) *Vychisl. Tekh. Vopr. Kibern.* **20** (1984) 121–135 (1984). Zbl 585.68009 ⇒

[18] A. Iványi, B. Novák, Testing of random sequences by simulation. *Acta Univ. Sapientiae Inf.* **2** (2) (2010) (megjelenőben).
http://compalg.inf.elte.hu/ tony/Kutatas/TAMOP/Papers-in-journals/.
⇒2

[19] A. Iványi, J. Pergel, Parallel processing of 0-1 sequences. *Ann. Univ. Sci. Budapest. Sect. Comput.* **4** (1983) 85–95. MR0750175 (85j:68039) 68Q10; Zbl 0541.68006, MSC2000: 68N25. ⇒3

[20] A. Iványi, J. Pergel, Performance evaluation of an algorithm, processing 0-1 sequences with priority. *Ann. Univ. Sci. Budapest. Sect. Comput.* **5** (1984), 37–40. MR0822597 (87g:68010a); Zbl 0613.65148, 65C99 60G35.
⇒3

[21] A. Iványi, J. Pergel, Bináris sorok párhuzamos kiszolgálása. (Hungarian) *Alkalmaz. Mat. Lapok* **11** (1–2) (1985) 191–200. MR0832285 (87g:68010b), 68N25 (90B22). ⇒1, 3

[22] I. Kátai, Új feladatok párhuzamos rendszerekben. Kézirat. ELTE IK, Budapest, 2010 október. ⇒3

[23] Novák Balázs: Sudoku algoritmusok elemzése. Programtervező matematikus diplomamunka. ELTE IK, Budapest, 2010.
http://compalg.inf.elte.hu/ tony/Oktatas/Rozsa/Sudoku-thesis/ ⇒2