計算機の時刻取得関数に関する性質とその解析法に関する考察

山口 健二[†]　中村 勝洋[‡]

[†] 千葉大学大学院自然科学研究科 数理物性科学専攻

[‡] 千葉大学大学院理学研究科 数学・情報数理学コース

1 はじめに

本稿では，計算機の時刻取得関数（clock_gettime() 関数）を使用して時刻を取得する際に成り立つ性質について述べる．ここではナノ秒レベルの高精度な時刻取得を想定している．特に取得時刻は，各計算機に依存した，ある一定の時間間隔の倍数でしか取得できないという性質を持つことを述べる．また，この時間間隔は，clock_gettime() 関数の分解能を得る clock_getres() 関数では求められないものであり，本稿では，クイックソートの実行時間分布を利用して，時刻取得関数の最小実行時間単位の算出法を導く．

2 計算機による時間管理

計算機にはリアルタイムクロック（Real Time Clock, RTC）と呼ばれる独立した回路が存在する．RTC には水晶発振器が内蔵されていて，発生する周波数を用いて現在の年月日や時間，分，秒を刻むことができる．しかし RTC に内蔵されている水晶発振器は安価なため，RTC を使ってナノ秒単位の高精度な時刻を刻むことはできない [1]．また，計算機内部の熱による周波数変動の影響を受けやすく，時刻に誤差が生じやすい．そのため，高精度な時刻を取得する際は，タイムスタンプカウンタ（Time Stamp Counter, TSC）と呼ばれる回路を用いる [2, 3]．TSC は CPU のクロック信号をカウントする 64 ビットのレジスタで，これを用いることで，計算機が動作しているときに限り，ナノ秒単位の高精度な時刻を刻むことができる．実際にナノ秒単位の時刻を取得するには，POSIX クロック[1]とそれに対応する関数を使用する方法がある．ナノ秒単位の時刻に関連する関数として clock_gettime() 関数と clock_getres() 関数などが存在する．clock_gettime() 関数を使うと，POSIX クロックの秒とナノ秒を取得することができる．また，clock_getres() 関数は，POSIX クロックの分解能を秒とナノ秒で取得する（1 ナノ以下の場合は 1 ナノ秒として取得される）．

ところで，clock_getres() 関数で得られた分解能が a ナノ秒だったとき，ある処理にかかる時間を clock_gettime() 関数を使って計測したとする．これを何度も行い実行時間の頻度を算出すると，実行時間は a ナノ秒間隔で分布すると一見思われる．しかし実際は異なり，実行時間は別の間隔で分布する．そこで，本稿では clock_getres() 関数では求めることができない clock_gettime() 関数による実行時刻の取得間隔の分布について詳細に調べる．そして clock_gettime() 関数を実行できる時刻間隔の最大公約数を最小実行時間単位 ω と見なして，その算出方法を導く．

表 1: 実験環境

	PC1	PC2	PC3	PC4
CPU	Intel Pentium4 2.80GHz	Intel Core2Duo 2.66GHz	Intel Core2Quad 2.40GHz	AMD Phenom 2.2GHz
Memory	2GB	2GB	2GB	2GB
OS	Fedora8 (32bit)	Fedora8 (64bit)	Fedora8 (64bit)	Fedora8 (64bit)

3 クイックソートを使った時刻取得関数に関する実験

clock_gettime() 関数を実行できる時刻間隔を調べるために，クイックソートを使った実験を行う．クイックソートを用いる理由は，ソーティングにかかる時間が前後し，ナノ秒レベルでの幅広い実行時間が得られ，解析し易くなることと，すでに時間計算量に関して理論的な考察が行われているため [5]，実験結果と比較できることである．

3.1 実験方法

4 台の計算機それぞれにおいて，clock_gettime() 関数によるクイックソートの実行時間を調べるプログラム（図 1）を実行した．各計算機の詳細は，表 1 のとおりである．図 1 の 4. の n はクイックソートされる要素数で，7. の R は実行時間の総数，D はこのプログラムに対する CPU 占有率の上昇が一段落するまでの 2.〜7. の部分のループ回数を示す．CPU 占有率が上昇中にはクイックソートの実行時間がある時を境に急に変動し安定しないため，CPU 占有率の上昇が一段落するまでは実行時間としてカウントしない．今回，$R = 10^9, D = 10^4, n = 32$ とし，clock_gettime() 関数で取得する POSIX クロックとして CLOCK_REALTIME を使用した．また，事前に clock_getres() 関数で CLOCK_REALTIME の分解能を調べたところ，PC1〜PC4 の計算機で 1 ナノ秒であった．これにより実験で使用した計算機の分解能は 1 ナノ以下であることが判明した．

4 時刻取得関数に関する性質 [6]

上記のプログラムを実行した場合，以下の性質が成り立つ．

A Note on Some Properties of clock_gettime() Function in a Computer and their Analysis Method

Kenji YAMAGUCHI[†] and Katsuhiro NAKAMURA[‡]

[†] Mathematical Sciences and Physics, Graduate School of Science and Technology, Chiba University.

[‡] Dept. of Math. & Informatics, Graduate School of Science, Chiba University.

kenji.yamaguchi@graduate.chiba-u.jp, nakamura@math.s.chiba-u.ac.jp

[1] POSIX クロックは複数の時計の総称で，その時計には CLOCK_REALTIME や CLOCK_PROCESS_CPUTIME_ID などがある．それらの時計では TSC で刻まれた時刻を使用できる [4]．（POSIX : Portable Operating System Interface for UNIX）

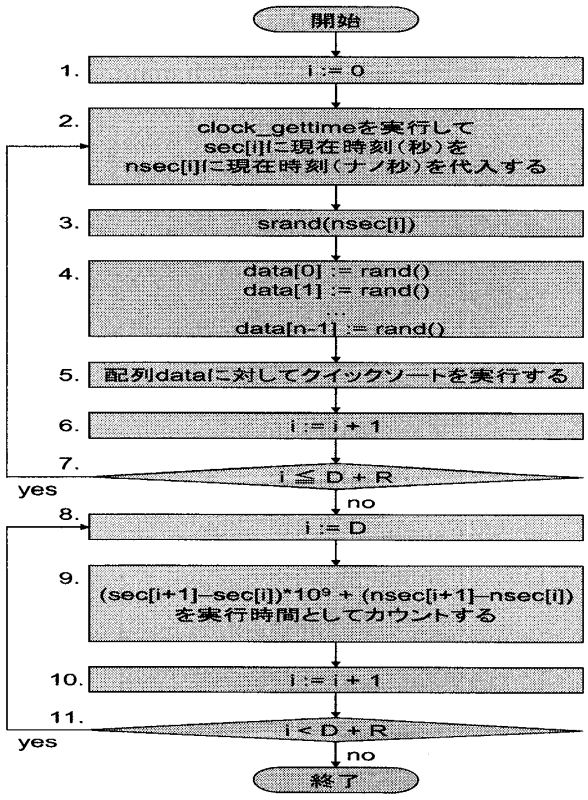

図 1: クイックソートの実行時間を調べるプログラム

【性質 1】実行時間 $m\omega$ に対し，

$$r_m = m\omega - \lfloor m\omega \rfloor \tag{1}$$

と定義する．このとき $\lfloor m\omega \rfloor = u - 1$ とおけば，clock_gettime() 関数で得られる実行時間の値は，確率 $1 - r_m$ で $u - 1$ となり，確率 r_m で u となる．特に ω が整数ならば $r_m = 0$ より，確率 1 で $u - 1$ となる．

★ 即ち，clock_gettime() 関数では，ある一定の時間間隔の倍数しか時刻を取得できないと述べたが，この時間間隔が実数なので，時刻を整数として取得する場合，同じ時間間隔であっても 2 通りの整数値が得られることがある．

【性質 2】実行時間の値 u を得る頻度を f_u とし，g_m を実行時間 $m\omega$ そのものの頻度とすれば，以下の式が成り立つ．

$$f_u = \sum_{m=\lceil (u-1)/\omega \rceil}^{\lceil u/\omega - 1 \rceil} r_m g_m + \sum_{m=\lceil u/\omega \rceil}^{\lceil (u+1)/\omega - 1 \rceil} (1 - r_m) g_m \tag{2}$$

ただし，$u = 0, 1, \ldots$ とし，$\lceil y \rceil$ は y を下回らない最小の整数を表し，$\ell < 0$ のとき $g_\ell = 0$ とし，$\ell > \ell'$ のとき，$\sum_\ell^{\ell'} r_m g_m = 0, \sum_\ell^{\ell'} (1 - r_m) g_m = 0$ とする．

★ 即ち，【性質 1】で述べた 2 通りの整数値が頻度 f_u を構成している（ω の値が大きいと右辺は，0 の場合もある）．

【性質 3】次の 2 つの式，

$$\lfloor m\omega \rfloor - \lceil (m-1)\omega \rceil \geq 1, \lfloor (m+1)\omega \rfloor - \lceil m\omega \rceil \geq 1 \tag{3}$$

が成立するとき，$\lfloor m\omega \rfloor = u - 1$ とおけば，

$$m\omega = u - 1 + \frac{f_u}{f_{u-1} + f_u} \tag{4}$$

表 2: 最小実行時間単位

	PC1	PC2	PC3	PC4
ω（ナノ秒）	3.758536	2.999882	3.745727	488.8147

となる．ただし，f_u は式 (2) で求められる頻度で，$f_{u-1} + f_u \neq 0$ とする．また $\omega < 2$ のときは $r_m \neq 0$ とする．

★ 即ち，ω の値が式 (3) を満たす大きさであれば，f_{u-1} と f_u の割合から $m\omega$ を算出することができる．

【性質 4】式 (3)，及び次の (5) 式

$$\lfloor (m+2)\omega \rfloor - \lceil (m+1)\omega \rceil \geq 1 \tag{5}$$

が成立するとき，$\lfloor m\omega \rfloor = u - 1, \lfloor (m+1)\omega \rfloor = v - 1$ とおけば，ω は次の (6) 式で算出できる．

$$\omega = v - u + \frac{f_v}{f_{v-1} + f_v} - \frac{f_u}{f_{u-1} + f_u} \tag{6}$$

★ 即ち，【性質 3】を m と $m+1$ の 2 箇所で用いることで，その差から ω を算出することができる．

ω の算出方法より算出した，各計算機の最小実行時間単位 ω を表 2 に掲載する（$(f_{u-1} + f_u)$ の値が最大となる部分を中心に 100 個（PC4 は 2 個）の ω 値を算出しその平均値である）[2]．すべての PC で $\omega \geq 2$ であり，PC4 のみ ω が大きな値になっていることがわかる．本章の方法で clock_gettime() 関数の最小実行時間単位 ω を得ることができ，性質をある程度解析できた．

5 まとめ

本稿では計算機の時刻取得関数に関する多くの性質を実験結果をもとに解析し，clock_gettime() 関数を実行できる時刻間隔が，ある一定の値を持つことを明らかにした．これは clock_getres() 関数など，計算機に最初から備わっている関数では調べられないものであることも確認した．そして，clock_gettime() 関数を実行できる時刻間隔の最大公約数（最小実行時間単位 ω）の算出方法を導いた．

参考文献

[1] 吉村一昭，倉持内武，安居院猛，図解入門よくわかる最新電波と周波数の基本と仕組み，秀和システム，2004

[2] J. Corbet, A. Rubini, G. Kroah-Hartman（著），山崎康弘他（訳），LINUX デバイスドライバ第 3 版，オライリー・ジャパン，2005

[3] D. P. Bovet, M. Cesati（著），高橋浩和（監訳），詳細 Linux カーネル第 3 版，オライリー・ジャパン，2007

[4] R. Love（著），千住治郎（訳），Linux システムプログラミング，オライリー・ジャパン，2008

[5] 溝井直史，尾崎俊治，“クイックソートの時間計算量の確率的解析，” 信学論（A），vol.J78-A，no.9，pp.1142–1148，Sept. 1995.

[6] 山口健二，中村勝洋，“計算機の時刻取得関数に関する性質とその解析，” 第 32 回情報理論とその応用シンポジウム予稿集，F14-3，pp.753–758，Dec. 2009.

[2] 計算機によっては，本稿で述べた算出方法を若干改良して ω の値を算出している部分もあるが，紙面の都合上省略する．