技 術 解 說

計算量 理論 概說

(1982年度 ACM Turing賞 受賞 記念 講演)

S. A. COOK**
林 濟 鐸* 譯
漢陽大學校 電子工學科 敎授(工博)*

要 約

計算量 理論을 역사적으로 槪觀하였다. 問題의 本質的 計算複雜度를 정의하는 것 및 計算量의 上界, 下界를 증명하는데 있어서의 기본적인 문제점에 중점을 두었다. 確率的 計算 및 並列計算에 관해서도 논하였다.

이 강연은 計算量(計算複雜度, computational complexity)理論에 관한 튜링賞 受賞 講演으로서는 두번째의 것이다. 첫번째 것은 1976년에 Michael Rabin의 것이다.① Rabin의 탁월한 記念講演[62]을 지금 다시 읽어보면 그 이래로 이 분야에 얼마나 많은 연구가 이룩되었는지를 절감하게 된다. 이 槪說에서는 計算量에 관한 硏究가 1960년에 시작된 이래 가장 중요하고 흥미를 끄는 결과라고 생각되는 문제에 대해서 언급하고자 한다. 이와같이 광범한 분야에 있어서는 話題의 선택이 개인적인 취향에 치우치는 것을 피할 수는 없지만 나로서는 어떠한 기준으로 보더라도 基本的이라 할 수 있는 論文은 포함시키도록 하겠다.

I. 初期의 論文

計算量 理論의 前史는 Alan Turing까지 거슬러 올라가야 할 것이다. Turing은 1937年의 論文「決定問題에 應用할 수 있는 計算可能한 數에 대해서, on computable numbers with an application to the entscheidungs problem」[65]에서 그의 유명한 튜링機械를 도입하였다. 튜링機械는 效果的으로(즉 알고리즘에 의해서) 計算할 수 있는 函數의 槪念에 관해서(그 당시로서는) 가장 설득력있는 形式化를 제공하였다. 일단 이 槪念이 정확하게 고정되면 計算機에 대한 計算不能性의 證明이 가능하게 된다. Turing은 그의 論文에서 다음을 증명하였다. 述語論理의 임의의 식이 주어졌을 때 그 식이 充足可能한가 여부를 유한회의 단계로 制定할 수 있는 알고리즘(즉 튜링機械)은 존재하지 않는다.

어떤 문제가 計算機로 풀 수 있는가 또는 풀 수 없는가를 설명하는 이론이 적절하게 개발된 연후에는 計算可能函數의 상대적인 計算困難 程度를 따지는 것이 자연스럽다. 이것이 이 講演의 主題인 計算量이다. Rabin[59,60]은 이 일반적 질문 "f가 g보다 計算하기 어렵다고 하는 것은 무엇을 뜻하는가"를 분명하게 표명한 최초의 사람들(1960)중의 하나이다. Rabin은 하나의 公理論的인 뼈대를 제안하였고 그것은 Blum[6]과 기타 사람들에 의해서 개발된 抽象的 計算量理論(abstract complexity theory)의 기초가 되었다.

큰 영향을 미친 初期(1965) 論文의 두번째 것으로는 J. Hartmanis와 R. E. Stearns[37]②의 論文 「알고리즘의 計算量에 관해서, On the computational complexity of algorithms」이다. 이 논문은 널리 읽혀졌으며 이 論文名이 바로 그 硏究分野의 명칭이 되었다. 그

Stephen A. Cook: "An Overview of Computational Complexity," CACM, Vol. 26, No. 6, pp. 401-408 (June 1983)

Stephen A. Cook 敎授의 주소 : Department of Computer Science, University of Toronto Toronto, Canada M5S 1A7.

① Michael Rabin과 Dane Scott가 공동으로 1976年度 튜링賞을 受賞하였다.

② Hartmanis[36]에서는 몇 가지 흥미있는 回想을 찾아 볼 수 있다.

論文에서는 多 테이프 튜링機械에 의한 計算時間을 사용해서 정의한 計算複雜度의 尺度라는 중요한 개념이 도입되고 階層定理(hierarchy theorem)가 증명되었다. 이 논문은 또 다음과 같은 오늘날까지도 未解決로 있는 흥미있는 問題를 제기하였다.

「임의의 非有理數(예를 들면 $\sqrt{2}$와 같은)를 實時間에 計算할 수 있는가? 즉 그 數의 10進表現을 100스텝 동작할 때마다 한 자리씩 언제까지나 프린트 해내는 튜링機械가 존재하는가?」

基礎를 굳히는 세번째의 論文(1965년)은 Alan Cobham[15]의 「函數의 本質的인 計算困難度, The intrinsic computational difficulty of functions」였다. Cobham은 "本質的(intrinsic)"이란 말을 강조하였다. 즉 그는 機械에 의존하지 않는 理論에 흥미를 가졌었다. 그는 곱셈이 보텔셈보다 어려운가 여부의 질문을 제기하고 이것을 다루는 理論이 적절히 개발되기까지는 그 質問에 답변할 수 없다고 믿었다. Cobham은 또 어떤 函數의 중요한 클래스를 정의하고 특징지웠다. 그가 α이라 부른 그 클래스는 自然數위의 函數로서, 그 計算時間이 入力의 10進 자리수의 어떤 多項式으로 제약되는 函數의 클래스였다.

위의 저자들 외에 또 다른 計算量理論 研究者들(나 자신을 포함해서)에게 영향을 준 다른 3개의 論文은 Yamada[91], Bennett[4], Ritchie[66]이다. 재미있는 것은 Rabin, Stearns, Benerett, Ritchie는 모두 거의 같은 시기에 Princeton大學의 학생이었다.

II. 初期의 論點과 概念

初期의 저자들중의 일부는 "計算複雜度는 무엇으로 測定하는 것이 옳은가"라는 문제에 관심을 가지고 있었다. 대부분의 사람은 당연한 선택으로서 計算時間 또는 領域量을 들었지만 그들이 유일한 또는 정확한 것이라고 확신하고 있지는 않았다. 예를 들면 Cobham[15]은 "일의 物理的 概念과 관련된 어떤 尺度가 가장 만족스러운 解析을 가져 올 것이다"라고 시사하였다. Rabin[60]은 計算量이 만족해야 할 公理를 도입하였다. 20年間에 걸친 경험을 통해서 나는 지금 時間과 領域 -특히 時間- 이 틀림없이 가장 중요한 計算量의 하나라는 것은 명백하다고 생각한다. 알고리즘의 效率을 評價하는데 중요한 제 1 의 數字는 그의 實行時間이라고 생각하는 것이다. 그러나 近年에 並列實行時間이나 하드웨어量도 또한 중요한 計算量의 尺度라는 것이 명백해지고 있다. (6. 參照)

적어도 Shannon[74] (1949年)까지 거슬러 올라가는 다른 또 하나의 중요한 計算量은 부울回路(組合論理回路)의 複雜度이다. 여기에서 편의상 문제의 函數 5는 有限長 비트列을 有限長 비트列로 高換하는 것이라 가정하고 f의 複雜度 C(n)을 길이 n의 모든 入力에 대해서 f를 計算하는 最小 부울回路의 크기라고 정한다. 이 매우 自然스러운 複雜度의 尺度는 計算時間과 밀접한 관계가 있으며([57a], [57b], [68b] 參照) 그 理論이 잘 개발되어 있다. (Savage[68a] 참조)

Cobham[15]에 제기한 또다른 질문은 計算過程에 있어서 1 "스텝"을 구성하는 것은 무엇인가라는 것이다. 이 질문은 알고리즘이 計算時間을 측정하기 위한 計算機의 올바른 모델은 무엇인가를 묻는 것과 같은 것이다. 多 테이프 튜링機械가 文獻에서는 가장 많이 사용되고 있지만 알고리즘의 效率的인 實現이라는 관점에서 볼 때 인위적인 제약을 가지고 있다. 예를 들면 記憶媒体가 直線狀의 테이프이어야 한다고 강제할 이유는 전혀 없는 것이다. 平面配列이나 木狀의 記憶으로는 안되는 것인가. 왜 랜덤 악세스 記憶은 허용하지 않는 것인가.

사실 1960년이래 計算機 모델의 제안이 꽤 많이 있었다. 현실의 계산기는 랜덤억세스 記憶을 가지고 있으므로 모델에 있어서도 이 기능을 허용하는 것이 자연스럽다고 생각한다. 그러나 그 기능을 어떻게 집어넣느냐 하는 것은 꽤 까다로운 문제이다. 만일 그 機械가 1스텝으로 整數를 스토어할 수 있다고 하면 그 수의 크기에 어떤 제한을 두지 않으면 안된다. (整數 2를 100번 제곱하면 결과는 2^{100}비트의 수가 되어 이세상에 존재하는 記憶媒体로는 스토어할 수 없을 것이다). 나는 文獻[19]에서 코스트付 랜덤억세스記憶(charged RAM)을 제안하였다. 그 記憶에서는 수 x를 저장하거나 또는 호출할 때마다 약 $\log|x|$만큼의 코스트(스텝數)를 부과한다. 이렇게 하면 되기는 되지만 그러나 썩 좋은 방법이라 하기는 어렵다. 보다 일반적으로 사용되고 있는 랜덤억세스모델은 Aho, Hopcroft, Ullman이 文獻[3]에서 사용한 것이다. 이 모델에서는 整數를 다루는 각 演算마다 단위 코스트가 걸리도록 되어 있으나 整數가 너무 커지는 것은 허용하지 않는다. (예를 들면 정수의 크기는 入力의 크기에 관한 어떤 多項式의 값으로 제한하는 등) 아마도 數學的으로 만족할만한 모델은 Schonhage의 記憶領域修正機械(storage modification machine)[69]일 것이다. 그 機械는 자기자신의 記憶構造를 구축하는 튜링機械라고 볼 수도 있고 또 1스텝으로 카피, 整數 1의 보텔셈, 또는 뺄셈, 저장 또는 呼出의 어느 하나를 실행할 수

있는 單一코스트 RAM이라고도 볼 수 있다. Schonhage의 機械는 훨씬 이전에(1958년) 제안된 Kolmogorov-Uspenski機械[46]를 약간 일반화합 것이다. Schonhage의 機械는 1스텝으로 限定된 양의 일을 하는 것으로 해석할 수 있는 가장 일반적인 機械를 나타내는 것이라고 생각된다. 곤란한 것은 약간 너무 강력한 것 같다는 것이다(3. 의 "큰수의 곱셈" 참조).

"1스텝이란 무엇인가"라는 Cobham의 質問에 대해서 이 20년간에 명백해진 것이 있다면 그것은 唯一하고 明確한 答은 없다는 것이다. 다행히 경합하고 있는 計算機모델은 모두 計算時間에 별 차이가 없다. 일반적으로 각 모델은 다른 모델에 의해서 計算時間을, 가장 긴 경우, 제곱하면 시뮬레이트할 수 있다. (이 결과에 대한 최초의 논의중의 몇개는 문헌[37]에 있다) 주류를 이루는 랜덤 억세스모델 중에서는 상호간 계산시간은 log의 率밖에 차이가 나지 않는다.

이와같은 議論의 결과 1965년까지에 하나의 중요한 최종적인 概念이 개발되었다. 즉, 入力長의 어떤 多項式으로 제한되는 計算時間내에 둘 수 있는 問題의 클래스를 認定하게 되었다. 多項式時間(polynomial time) 알고리즘과 指數的時間(exponential time) 알고리즘은 이미 1953년에 Von Neumann[90]에 의해서 구별이 되었었다. 그러나 그 클래스는 Cobham[15]이 1964년에 函數의 클래스 α(1. 참조)을 도입할 때까지는 形式的으로 定義되지도 않았으며 연구되지도 않았었다. Cobham은 그 클래스가 명확하게 定義되며, 사용하는 컴퓨터 모델에 의존하지 않는 것을 지적하고 그 클래스를 再歸函數理論에서 하는 것과 같은 방법으로 특성화 하였다. 多項式時間計算可能性은 대충 處理容易度(tractability)에 대응한다는 생각은 Edmonds[27]에 의해서 최초로 인쇄물로서 발표되었다. 그는 多項式時間알고리즘을 "좋은 알고리즘(good algorithm)"이라 불렀다. 多項式時間으로 인식할 수 있는 系列集合의 클래스를 나타내는 새로운 표준적인 記法P는 뒤에 Karp[42]에 의해서 도입되었다.

P를 處理용이한(실제로 계산할 수 있는(feasible)) 문제와 同一視하는 일은 1970년대의 초기이래로 그 研究分野에서는 일반적으로 인정되어 왔었다. 그러나 그것이 왜 그렇게 되는지가 곧바로 명백하지는 않았다.

왜냐하면 實行時間이 多項式 n^{1000}인 알고리즘은 확실히 실제로는 실행 불가능이고 또 역으로 實行時間이 指數 $2^{0.00DIN}$인 알고리즘은 실제적으로 실행이 가능하기 때문이다. 그러나 자연적으로 발생하는 문제는 그와 같은 실행시간을 가지는 最適 알고리즘을 갖지 않는다는 것이 경험상의 사실인 것 같다.③ 최악의 경우의 실행시간이 指數的이 되는 가장 두드러진 실용적 알고리즘은 線型計劃問題에 대한 심플렉스 알고리즘이다. Smale[75, 76]은 어떤 의미에서 그 알고리즘의 平均實行時間이 빠르다라는 것을 보임으로서, 실용적 알고리즘이라는 것을 설명하려고 하였다. 그러나, Khachian[43]은 다른 알고리즘을 사용하여 線型計劃問題가 P에 속한다는 것을 증명하였다는 사실에 주목할 필요가 있다. 이리하여 우리의 一般的主張－P는 실제적으로 풀리는 문제와 일치한다－는 위배되지 않았다.

Ⅲ. 計算時間의 上界

計算機科學의 중요한 연구분야로서 많은 效率이 좋은 알고리즘을 설계하고 해석하는 것이 있다. (計算複雜度의 관점에서 볼 때) 중요한 알고리즘은 특수한 것임에 틀림없다. 그들은 간단 또는 중요한 문제를 풀기 위한 놀라울 정도로 빠른 방법을 제공한다. 이제 1960년 이후에 고안된 흥미있는 알고리즘중에서 몇 가지를 들기로 한다. (私見이지만, 무엇이 언제나 가장 중요한 알고리즘인가에 대해서 사색하는 것은 흥미있는 일이다. 10進數에 대한 산술연산 +, −, ×, ÷는 기본적인 것이다. 거기에 따르는 것으로는 高速分類 및 探索, 가우스 消去法, 유크리드互除法, 심플렉스 알고리즘을 그 후보로 들 수 있다)

파라미터 n은 入力의 크기를 나타내고 計算時間의 上界는 最惡의 경우의 計算時間의 上界이며 多 테이프 튜링機械(또는 다른 타당한 랜덤억세스 機械)에 적용되는 것으로 한다. (별도로 말하지 않는 경우는)

1. 高速푸리에 變換

高速푸리에 變換[23]은 $0(n \log n)$回의 算術演算을 實行하며 科學計算에서 가장 많이 사용되고 있는 알고리즘중의 하나이다.

2. 큰수의 곱셈

국민학교에서 배우는 방법으로는 두 n자리 수의 곱셈에는 $0(n^2)$回의 비트演算을 실행한다. 1962년에 Karatsuba와 Ofman[41]이 단지 $0(n^{\log 3})$回의 연산으로 되는 방법을 발표하였다. 그 바로 후 Toom[84]은 곱셈을 실행하는 부울回路로 임의의 작은 ε에 대해서 $0(n^{1+\varepsilon})$인 크기를 가지는 回路의 構成法을 보였다. 그 당시 Harvard大學의 大學院生이었던 나는 Cobham의 質問 "곱셈은 보텔셈보다 어려운가?"에 많은 고무를

③이에 관한 논의는 文献[31] pp. 6-9를 참조할 것.

받았다. 나는 순진하게 곱셈은 多 테이프 튜링機械로 $\Omega(n^2)$回의 스텝이 필요하다는 것을 증명하려고 시도하고 있었기때문에 100m의 論文은 나를 크게 놀라게 하였다. 나는 Stal Aanderaa[22]의 도움을 받아 곱셈은 "온라인" 튜링機械를 사용 $\Omega(n \log n/(\log \log n)^2)$回의 스텝이 필요하다는④ 것을 보였다. 나는 또 學位論文속에서 Toom의 방법들 多테이프 튜링機械에 응용해서 $0(n^{1+\epsilon})$回의 스텝으로 곱셈이 된다는 것-Toom에게는 놀라운 일이 아닌 이미 해 오던 일이겠지만-을 지적하였다.

수의 곱셈에 대한 多테이프 튜링機械에 의한 현재 가장 高速의 漸近的 實行時間은 $0(n \log n \log \log n)$이며 그 방법은 高速푸리에 變換을 이용해서 Schönhage와 Strassen[70]이 고안하였다(1971년). 그러나 Schönhage[69]는 최근 그의 記憶域修正機械(2. 참조)가 $0(n)$時間(n에 비례하는 시간!)으로 곱셈을 실행할 수 있다는 것을 복잡한 의론에 의해서 보였다. 우리는 지금 곱셈이 생각했던 것보다 수월한 것인지 아니면 Schönhage의 機械가 우리를 기만하는 것인지에 대한 결론을 내려야 할 입장에 있는 것이다.

3. 行列의 곱셈

잘아는 방법은 두 $n \times n$行列을 곱하는 데 $n^2(2n-1)$回의 算術演算을 요한다. 1950년대와 1960년대에는 그 방법이 실은 최적이라는 것을 증명하려고 시도하였었다. Strassen[81]이 단지 $4.7n^{2.81}$回의 演算으로 실행하는 방법을 발표했을때(1969년) 모두들 놀랐었다. 2.81이란 지수를 감소하기 위해서 꽤 많은 연구가 행해졌으며 현재 알려져 있는 가장 좋은 방법의 시간上界는 Copper Smith와 Winograd[24]에 의한 $0(n^{2.496})$이다. 알려져 있는 最良의 下界는 $2n^2-1$(文獻[13] 참조)이므로 아직 進步의 여지는 충분이 있다 하겠다.

4. 一般無何 그래프의 最大맷칭

이 문제는 P에 속한다는 것이 陽的으로 증명된 문제중에서 P에의 소속을 표시하는데 어려운 알고리즘을 필요로 하는 최초의 문제였다. Edmonds의 영향력있는 論文[27]은 그 결과를 보였고 多項式時間 알고리즘(2. 참조)의 개념을 논하였다. 그는 또 擴大經路(augmenting path)라는 단순한 개념-그것은 2分 그래프의 경우는 충분했었지만-은 일반 無向그래프에 대해서는 적용되지 않는다는 것을 지적하였다.

④이 下界는 약간 개선되었다. 文獻[56],[64] 참조.

5. 素數의 判定

여기에서 주된 논점은 이 판정문제가 P에 속하는가 여부이다. 다시 말하면 임의의 n자리 정수를 주었을 때 그것이 소수인가 아닌가를 언제나 정확하게 판정하고 또 n의 어떤 고정된 多項式으로 제한되는 스텝數 이내에 停止하는 알고리즘이 있는가라는 것이다. Gary Miller[53]는 그와같은 알고리즘이 존재한다는 것을 보였지만(1976년) 그 정당성은 擴張리만仮定(extended riemann hypothesis)에 의존하고 있다. Solovay와 Strassen[77]은 素數를 판정하기 위한 高速 몬테칼로(monte carlo) 알고리즘(5. 참조)을 고안했지만 入力數가 合成數일 때(素數가 아닐 때)에는 그 알고리즘이 잘못해서 소수라고 해버릴 가능성이 약간 있다. 알려져 있는 決定性 알고리즘에서 가장 좋은 것은 Adleman, Pomerance 및 Rumely[2]에 의해서 고안된 것으로 $n^{0(\log \log n)}$ 시간 걸린다. 이 시간은 多項式時間보다 약간 나쁘다. H. Cohen과 H. W. Lenstra Jr.[17]에 의해서 제안된 이들의 變形數 알고리즘은 10進 100자리까지의 수를 약 45초이내에 루우틴적으로 처리할 수가 있다.

최근에 3개의 중요한 문제가 클래스 P에 속한다는 것이 밝혀졌다. 첫번째 것은 1979년에 Khachian[43]에 의해서 밝혀진 線形計劃問題이다(解說은 文獻[55] 참조). 두번째 것은 1980년에 Luks[50]에 의해서 밝혀진 次數가 최고 d인 두 그래프의 同型判定問題이다(그 알고리즘의 計算時間은 d를 고정하면 頂點數의 多項式이지만 d에 대해서는 指數的이다). 제3의 것은 有理係數 多項式의 因數分解이다. 이것은 1982년에 A. Lenstra, H. Lenstra 및 Lovasz[46]에 의해서 1變數 多項式에 대해서 밝혀진 것이다. 그 방법은 Kaltofen의 결과[39],[40]에 보인 바와 같이 임의의 고정된 개수의 변수를 가지는 多項式으로 一般化할 수 있다.

Ⅳ. 下 界

計算量理論에 있어서의 진짜 挑戰-그 理論을 알고리즘 解析과 分離시키는 역할을 한 문제-는 특정문제의 計算量의 下界를 證明하는 것이다. Yes-no 問題가 어떠한 알고리즘을 사용하더라도 n 또는 n^2, 또는 2^n 스텝으로 절대로 풀리지 않는다는 것을 증명하는데 사용할 만족할만한 무엇인가가 있다. 下界를 증명한 몇가지 중요한 성공사례도 있지만 未解決의 問題가 더욱 중요하고 또 다소 실망적이기도 하다.

計算時間 또는 記憶領域量의 모든 중요한 下界는 "對角線論法"에 바탕을 두고 있다. 對角線論法은 Tu-

ring과 그 同時代의 研究者들이 어떤 問題가 알고리즘的으로 풀 수 없다는 것을 증명하는데 사용하였다. 對角線論法은 또 計算可能한 0－1函數⑤의 階層構造를 정의하는데 사용하였다. 1960년에 Rabin[60]은 합리적인 어떠한 計算量 尺度(예를 들면 計算時間 이라든가 記憶領域)를 사용하는 허울 計算時間 또는 記憶領域 등을 충분히 증가시키면 항상 보다 많은 0－1函數를 計算할 수 있게 된다는 것을 증명하였다. 거의 같은 時期에 Ritchie는 그의 學位論文[65]에서 許容記憶領域量에 의해서 函數의 어떤 特定階層構造(그가 보인 바와같이 0－1函數에 대해서 自明하지 않다.)를 정의하였다. 얼마뒤에 Rabin의 結果는 Hartmanis와 Stearns[37]에 의해서 多테이프 튜링機械의 計算時間에 대해서 또 Stearns, Hartmanis, Lewis[78]에 의하여 記憶領域量에 대해서 자세하게 擴張되었다.

1. 實際的計算不可能이라고 證明된 自然的 判定問題

위의 階層構造에 관한 결과는 특정 函數를 계산하는데 필요한 計算時間과 記憶領域量에 대한 下界를 주기는 했지만 그들 函數는 모두 "운좋게 해치운" 듯한 감이 든다. 예를 들면 機械 x에 入力 y를 주었을 때 $(|x|+|y|)^3$ 스텝후의 出力의 첫자리수를 주는 函數를 f(x, y)라 하면 용이하게 알 수 있는 바와 같이 f(x, y)는 $(|x|+|y|)^2$時間 이내에는 계산할 수 없다. "自然"문제에 대한 일반적 計算모델에 있어서 自明하지 않은 下界(흥미가 있고 또 計算機械에 관계하고 있지 않다는 의미에서 自然)가 발견된 것은 1972년이 되어 Albert Meyer와 Larry Stockmeyer[52]가 자승연산을 가지는 正規表現의 等價性判定問題는 指數的領域, 따라서 指數的計算時間을 필요로 한다는 것을 증명했을 때이다. 그보다 약간 뒤에 Meyer[51]는 WSIS (後者函數를 가지는 弱單項 2階論理, Weak monadic second-order theory of successor)라 불리는 決定可能한 形式的体系에 있어서의 論理式의 眞僞 判定에 필요한 計算時間의 매우 強한 下界를 발견하였다. 그는 크기가 정해진 指數(2^n, 2^{2n}, $2^{2^{2n}}$ 등)로 제한되는 計算時間으로는 어떠한 計算機械도 WSIS問題를 바르게 判定할 수 없다는 것을 증명하였다. Meyer의 Ph. D. 학생이었던 Stockmeyer는 그 일을 계승해서 다음과 같은 계산을 하였다.[79] "길이 616記號의 임의의 WSIS論理式의 眞僞를 바르게 判定하는 어떤 부울回路(계산기를 생각하면 된다)도 10^{123}個보다 많은 게이트를 갖어야 한다". 10^{123}이란 수는 宇宙에 꼭 들어가는 陽子의 수가 되도록 선택하였다. 이것이야 말로 납득할 수 있는 實行不可能性의 證明이 아닌가!

Meyer와 Stockmeyer이래 決定可能한 形式的体系의 計算量의 下界가 많이 발견되었다(要約은 文献[29],[80] 참조). 가장 흥미있는 것중의 하나는 Fisher와 Rabin[30]에 의한 푸레스버거算術(Presburger arithmetic; 보텔셈에 의해서 닫혀진 自然數의 理論体系)의 眞僞判定에 요하는 計算時間의 2重指數(2^{2^n})時間下界이다. 이 下界는 이 理論体系에 대한 알려져 있는 最良의 時間計算量上界[54] – 3重指數($2^{2^{2^n}}$)時間 – 와 그다지 차이가 없다. 알려진 最良의 領域計算量上界는 2重指數이다.[29]

위와같은 성공에도 불구하고 보다 작은 計算量을 가지는 문제에 대한 下界를 證明한 기록은 거의 없다. 실제로 NP(4.4 참조)에 속하는 임의의 自然問題 특히, 文献[31]에 나열되어 있는 300개나 되는 문제의 어느 하나도 汎用計算모델에 의한 非線型時間(non linean time)下界가 발견되지 않았다. 물론 對角線論法에 의해서 임의의 정해진 k에 대해서 n^k時間을 요하는 문제가 NP중에 존재한다는 것은 증명된다. 그러나 領域計算量下界의 경우에는 오프라인 튜링機械(4.3 참조)로 0 (log n)의 作業領域에서 풀 수 없는 문제가 NP안에 존재한다는 것을 證明하는 방법조차 우리는 모르고 있다. 많은 自然의 경우 알려진 最良의 領域計算量上界가 본질적으로 n에 비례한다는 사실에도 불구하고 그러하다.

2. 構造化下界(Structured Lower Bound)

일반 計算機모델을 사용하여 구체적 문제에 대한 計算量下界를 證明하는 일에는 거의 성공을 했지만 "構造化(struclured)"모델에 대해서는 흥미있는 결과가 나와 있다. "構造化" 또는 말은 문제에 적합한 한정한 操作만을 갖춘 計算機를 지칭하기 위해서 Borodin[9]이 導入하였다. 構造化下界가 구해진 간단한 예는 n個의 數를 分類하는(大小順으로 재배열하는)문제이다. 이 문제는 허용된 計算機의 演算은 入力되는 두 수의 大小를 비교하는것 뿐이라는 전재하에서 적어도 n log n번의 大小比較를 요한다는 것을 증명하는 것은 그다지 어렵지 않다(文献[44] 참조). 이 下界는 튜링機械나 부울回路에 관해서는 언급이 없지만 나눌셈이 허용되지 않는다는 전재하에 單一코스트 랜덤억세스 機械로 擴張되어 있다.

⑤예를 들면 Grzegorczyk[35] 참조.

제 2 의, 그리고 우아한 構造化下界는 Strassen[82] (1973년)에 의한 것인데, 다음과 같다. 多項式補間問題, 즉 주어진 n개의 점을 지나는 n−1次 多項式의 係數를 구하는 문제는, 算術演算만이 허용된다고 할 때 Ω(n log n)번의 곱셈을 요한다. 여기에서 재미있는 것은 Strassen의 최초의 증명이 代數幾何(algebraic geometry)의 심오한 結果인 Bezout의 定理에 바탕을 두고 있다는 것이다. 아주 최근에 Barr과 Strassen[83]은 그 下界를 擴張해서 n點을 지나는 多項式의 係數 중 中間次數의 것만을 구하는데도 Ω(n log n)번의 곱셈이 필요하다는 것을 보이었다.

이를 모든 構造化下界에 관한 結果의 매력은 그 下界가 알려진 최량의 上界에 가깝고⑥ 또 최량의 알려진 알고리즘이 그 下界가 적용되는 構造化機械로 實現할 수 있다는 것이다 (基底分類(radix sort)는 때로 線形時間으로 實行할 수 있다고 하지만 入力數의 자리수가 충분히 크고 數가 모두 다르다고 가정할 수 있다면 실제에는 적어도 n log n스텝이 필요하다는 점에 주의하기 바란다).

3. 時間-領域 곱의 下界

時間計算量이나 領域計算量의 下界를 증명하는 곤경에서 빠져나오는 또 하나의 방법은 작은 領域밖에 사용할 수 없다는 가정하에서 時間計算量의 下界를 증명하는 것이다. Cobham[16]은 그와 같은 최초의 결과를 1966년에 증명하였다. 그는 그때 주어진 n자리의 수가 完全平方數인가 여부를 "오프라인" 튜링機械로 식별하기 위해서는 時間-領域곱이 Ω(n²) 이상이 되지 않으면 안된다는 것을 증명하였다. (동일한 사실이 길이 n 인 記號列의 回文(palindrome)의 식별에 대해서도 성립한다. 여기에서 入力은 2방향 읽기전용 入力테이프 위에 쓰고 사용된 領域이란 정의에 의해서 그 튜링機械가 사용할 수 있는 작업용 테이프 위에서 走査된 네모수이다. 따라서 만일 예를 들어 領域量을 0 (log³n) (이것은 충분할 정도 이상으로 크다) 이하로 제한하면 計算時間은 Ω(n²/log³n) 이상이 되지 않으면 안된다)

Cobham의 결과의 약점은 오프라인 튜링機械모델을 사용하고 있다는 것이다. 오프라인 튜링機械는 計算時間이나 作業領域量을 따로 따로 측정하는데 적합한 모델이지만 計算時間과 領域을 동시에 생각할 때에는 너무 制約이 심하다. 예를 들면 回文은 만일 2개의 헤드로 入力테이프를 동시에 주사하는 것을 허용한다면 명백히 2n스텝으로, 그리고 또 一定領域量으로 식별할 수 있다. Borodin과 나[10]는 1에서 n²까지의 범위에 드는 n개의 整數를 分類하는 데는 Ω(n²/log n)의 時間-領域곱이 필요하다는 것을 증명하고 그 약점을 부분적으로 개선하였다. 그 증명은 임의의 "一般順序機械(general sequentiel machine)에도 적응할 수 있다. 그 機械모델은 入力테이프에 複數個의 入力헤드, 또는 랜덤억세스機能까지 있는 오프라인 튜링機械를 포함한다. 불행하게도 그 증명에 결정적인 것은 分類에 많은 출력비트를 필요로 한다는 것이다. 비슷한 下界가, 예를 들면 주어진 n개의 수가 모두 다른가 여부를 식별하는 것과 같은 集合認識問題(set recognition problem)에 적용하도록 할 수 있는가는 흥미있는 未解決의 問題로서 남아있다. (分類에 관한 우리의 下界는 최근 文獻[69]에서 약간 개선되었다)

4. NP 完全

NP 完全의 理論은 확실히 計算量理論에 있어서 가장 중요한 發展이다. 나는 지금 여기에서 그 理論에 관해서 장황하게 논하려고 하지는 않는다. 왜냐하면 그것은 이제 잘 알려진 사실이고, 教科書에서 다룬 主題이기 때문이다. 특히 Garey와 Johnson의 책[31]은 그에관한 탁월한 책이다.

클래스 NP는 非決定性 튜링機械로 多項式時間내에 認識할 수 있는 集合으로 되어 있다. 내가 아는 한 수학적으로 등가인 클래스가 최초로 정의된 것은 1962년 James Bennett의 Ph. D. 學位論文[4]에서이다. Bennett는 그의 클래스를 나타내는데 "擴張陽初學關係(extended positiae rudimentary relation)"라는 명칭을 사용하고 그의 定義에서는 計算機 대신 論理的限定作用事(logical quantifier)를 사용하였다. 나는 그의 學位論文의 이 부분을 읽고 그의 클래스는 지금 잘 알려져 있는 NP로서 特性化할 수 있다는 것을 이해하였다. 나는 1971년의 論文[18]에서(Cobham의 클래스 α에 따라) α^+라는 용어를 사용하였다. 그리고 Korp는 1972년 그의 論文[42]에서 지금 사람들이 받아 들이고 있는 이름 NP를 그 클래스에 부여하였다. 한편 形式的議論의 발전과는 전혀 관계없이 Edmonds는 1965년에[28] "좋은 特性化(good characterization)"를 가지는 문제에 관해서 非形式的으로 논하였다. 그 개념은 本質的으로는 NP와 같은 것이다.

1971년[18]에 나는 NP完全(NP-complete)의 槪念을 도입하고 3-充足可能性(3-satisfiability) 問題와 部分그래프問題가 NP完全임을 證明하였다. 1년후 Ka-

⑥ 多項式 補間問題의 上界에 관해서는 Borodin과 Munro[12]참조.

rp[42]는 21개의 문제가 NP完全임을 증명함으로서 이 문제의 중요성을 과시하였다. 이와는 독립적으로 약간 뒤에 소련(현재는 보스톤大學)의 Leonid Levin[49] 은 비슷한(보다 강한) 概念을 정의하고 6개의 문제가 그의 의미에서 完全임을 증명하였다. "探索問題(search problem)"의 非形式的 概念은 소련의 文獻에서는 표준적이고 Levin은 그의 問題를 "普遍探索問題(universal search problem)"라 불렀다.

클래스 NP는 비지니스나 인더스트리에서 발생하는 실제적인 문제를 매우 많이 포함하고 있다. (文獻[31] 참조) 만일 모든 NP問題가 P에 속하지 않으면 어떤 NP問題가 NP完全비라는 것을 증명하려면 그 問題가 P에 속하지 않는다(決定性 多項式 時間 알고리즘을 갖지 않는다)는 것을 증명하는 것이 된다. 앞의 조건 「모든 NP問題가 P에 속한다」는 것이 만일 사실이라면 計算機科學에 大革命을 유발하는 것이므로 NP 完全하다는 것의 실용상의 효과는 「下界」를 보인 것이다. 이것이 곧 이 硏究主題를 下界에 관한 節에 포함시킨 이유이다.

5. #P 完全

NP完全의 개념은 集合에 대한 것이며 어느 集合이 NP完全이라는 증명은 통상 그 集合(의 認識)이 처리하기 어렵다(intractoble)는 것의 證明이라 해석된다. 그러나 NP完全性의 증명이 적합하지 않으면서 처리하기 어렵다는 것이 분명한 函數가 많이 있다. Lestie Valiant[86, 87]는 이와같은 상황을 구제하기 위해서 #P完全(#P-completeness)이라는 개념을 정의하였다. 어떤 函數가 #P完全하다는 것을 증명하는 것은 마치 어떤 集合의 NP完全性의 증명이 그 集合의 認識이 분명히 처리하기 어렵다는 것을 나타내는 것과 마찬가지로 그 函數의 計算이 분명히 처리하기 어렵다는 것을 나타내는 것이다. 즉 만일 #P完全인 函數가 多項式時間에 計算可能하면 P=NP가 성립한다.

Valiant는 #P完全인 函數의 많은 例를 들었다. 그 중에서 가장 흥미있는 것 중의 하나는 아마도 整數行列의 퍼머넨트(permanent)를 구하는 函數일 것이다. 퍼머넨트는 行列式과 形式的으로 아주 닮은 定義를 가지고 있지만 行列式이 가우스 消去法에 의해서 용이하게 계산되는데 대해서 퍼머넨트를 계산하는 실제적으로 實行可能한 방법을 발견하려는 과거 百有餘年에 걸친 많은 시도는 모두 실패하였다. Valiant는 퍼머넨트를 구하는 函數는 #P完全임을 證明함으로서 그 실패에 대한 납득할만한 최초의 理由를 들었다.

V. 確率的 알고리즘

確率過程을 시뮬레이트 또는 近似하는데 亂數를 사용하는 것은 매우 자연스럽고 실제면에 있어서도 잘 確立되어 있다. 그러나 決定論的인 組合問題를 할 때 랜덤人力이 有效할 것이라는 생각이 計算機 科學社會에 침투하는 속도는 매우 느리었다. 여기에서 나는 그것을 푸는 決定性 多項式時間 알고리즘이 알려져있지 않은 문제를(적당한 의미에서) "푸는"確率的(동전 던지기(coin tossing)식) 多項式時間 알고리즘에 한정해서 생각해 보기로 한다.

그와같은 최초의 알고리즘은 1970년의 Berlekamp[5]에 의한 P개의 원소를 가지는 体 GF(P) 위에서 多項式 f를 因數分解하기 위한 것이라고 생각된다. Berlekamp의 알고리즘은 f의 次數와 log P의 多項式時間으로 실행하며 적어도 確率 1/2로 f의 바른 素因數分解를 발견한다. 그렇지 않으면 失敗해서 終了한다. 그 알고리즘은 몇번이던지 되풀이해서 實行할 수 있으며 또 失敗事象도 모두 서로 獨立이므로 그 알고리즘은 사실상 항상 實際的으로 可能한 計算時間內에 因數分解를 행한다.

보다 極的인 예는 Solivay와 Strassen[77] (1974년에 投稿)에 의한 素數判定 알고리즘이다. 이 알고리즘은 入力으로 주어지는 數 m의 자릿수의 多項式時間에 實行하며 "素數" 또는 "合成數"의 어느 하나를 出力한다. m이 실제로 素數이면 出力은 분명히 "素數"이지만 m이 合成數일 경우에는 최고 1/2의 확율로 답이 "素數"로 될 가능성이 있다. 어떤 入力 m에 대해서 그 알고리즘은 독립적인 出力結果를 얻으면서 몇번이던지 되풀이해서 실행할 수 있다. 따라서 만일 답이 "合成數"로 나오기만 하면 알고리즘 이용자는 m이 정말로 合成數임을 알게 된다. 또 그 답이 예를 들어 100回의 실행에 대해서 언제나 "素數"라 나오면 m이 素數라는 확신을 얻는다. 그것은 임의의 고정된 合成數 m이 그와 같은 出力結果를 줄 확율은 매우 작기(2^{-100} 보다 작다)때문이다.

Rabin[61]은 위의 성질과 비슷한 성질을 가지는 다른 確率的 알고리즘을 개발하였다. 計算機로 시험해 본 결과 매우 高速이었고 $2^{400}-593$은 數分이내에 素數로 식별되었다.

確率的 素數 判定法의 흥미있는 하나의 應用은 Rivest, Shamir, Adleman[67a]에 의해서 1978년에 출판된 公開鍵暗號系(pubcickey cryptosystems)에 관한 획기적인 논문에서 제안되었다. 그들의 暗號系는 큰

(100자리 정도의) 랜덤 素數의 발생을 필요로 한다. 그들은 랜덤으로 택한 100자리의 수를 Solovay-Strassen의 방법을 사용해서 위에서 略述한바와 같은 의미에 있어서 素數일 것이라고 알 때까지 테스트할 것을 제안하였다. 실제로는 랜덤인 100자리의 "소수일 공산이 큰수"가 일단 발견되었을 때 만일 틀림없이 素數라는 것을 확인할 필요가 있으면 3에서 설명한 Cohen과 Lenstra[17]의 새로운 高性能 決定性 素數判定 알고리즘을 사용하여 약 45초 걸려서 정말 素數인가 여부를 테스트할 수 있을 것이다.

Solovay와 Strassen의 의미에서 多項式時間確率的 認識 알고리즘을 가지는 集合의 클래스는 文獻에서는 R(또는 때때로 RP)로서 알려져 있다. 따라서 어떤 集合 S가 R에 속한다고 하는 것은 S가 다음과 같은 確率的 認識 알고리즘을 가질 때 그리고 그때에 한한다. 그 알고리즘은 항상 多項式 時間에 정지하고 또 R에 속하지 않는 入力系列에 대해서는 결코 과오를 범하지 않고, 또 R에 속하는 入力系列에 대해서는 每 實行마다 적어도 1/2의 확률로 바른 답을 출력한다. 따라서 合成數 集合은 R에 속하고 또 일반적으로 P⊆R⊆NP이다. R에는 속하지만 P에 속하는지는 모르고 있는 흥미있는 集合의 예도 있다. 예를 들면 Schwartz[71]는 다수의 변수를 가지는 多項式을 要素로 가지는 正則行列의 集合은 R에 속한다는 것을 보이었다. 그 判定 알고리즘은 랜덤으로 택한 작은 整數값에 대해서 각 多項式을 평가하고 그 결과의 行列式을 計算한다 (多項式을 계산하면 일반적으로 指數的인 많은 항을 가질 수 있기 때문에 直接 行列式을 계산하는 것은 실제적으로 不可能하다는 것이 명백하다).

R=P인가 여부는 흥미를 끄는 未解決問題이다. 구하고 있는 답이 잘 定義된(well defined) "yes" 나 "no"일 경우 무작위의 동전 던지기(random coin tosses)는 별로 쓸모가 없을 것이다. 나는 哲學的 基盤을 바탕으로 R=P라고 豫測하려는 쪽으로 마음이 끌린다. 하나의 관련된 問題는(어떤 문제가 R에 속하는 것을 보이는) 確率的 알고리즘은 決定性 알고리즘과 마찬가지로 모든 實用的 目的에 대해서 효과적인가라는 것이다. 결국 確率的 알고리즘은 대부분의 計算機에서 이용할 수 있는 擬似亂數發生機能을 사용해서 실행할 수 있으며 또 2^{-100}의 誤 確率은 무시할 수 있다. 결점은 擬似亂數發生 루틴은 참된 亂數를 발생하지 않는다는 것이다. 擬似亂數가 주어진 確率的 알고리즘에 대해서 얼마나 잘 들을 것인가는 아무도 모른다. 실제 경험에 의하면 그들이 잘 듣는 것처럼 보인다. 그러나 그것이 항상 잘 듣는다면 R=P이다. 왜냐하면 擬似亂數는 決定論的으로 발생되며 랜덤性은 결국 아무 도움도 되지 않했기 때문이다. 亂數를 발생하기 위한 다른 가능성은 熱雜音과 같은 物理現象을 이용하는 것이다. 그러나 自然은 정말로 랜덤한 것인가라는 것은 科學의 原理에 있어서의 未解決 問題이다.

클래스 R에 관한 Adleman[1]의 흥미있는 定理에 관해서 언급하고 이 절을 마치기로 한다. 용이하게 알 수 있는 일인데[57b] 만일 어떤 集合이 P에 속한다면 任意의 n에 대해서 n의 어느 固定된 多項式으로 그 크기가 제한되는 부울回路로, 길이가 n인 任意의 入力系列에 대해서 그것이 그 集合에 속하는지 여부를 決定하는 부울回路가 존재한다. Adleman이 증명한 것은 클래스 R에 대해서도 그것이 성립한다는 것이다. 따라서 예를 들면 각 n에 대해서 주어진 n자리의 수가 素數인가 여부를 바르고 신속하게 테스트하는 小規模의 "計算機回路"가 존재한다. 결점은, 그 回路는 n에 대해서 一樣하지 않고 또 실제로 100자리의 경우에도 그 回路를 어떻게 構成할 것인가를 계산해 내는 것 자체도 현실적으로 實行이 不可能할는지도 모른다.⑦

Ⅵ. 同期並列計算

VLSI技術의 출현으로 1/4인치 칩에 1개 또는 복수개의 프로세서를 만들 수 있게 되었으므로 하나의 문제를 풀기 위해서 수 1000개의 그와 같은 프로세서를 並列로 構成하여 동시에 동작시키는 未來의 計算機에 대해서 생각하는 것은 자연스러운 일이다. 이와같은 류의 大型 汎用計算機가 아직 만들어지지는 않았지만 진행중인 프로젝트는 있다(Schwartz[72] 참조). 이와 같은 동기에 의해서 計算量理論의 하나의 재미있는 분야가 최근 발달해 왔다. 그것은 大規模 同期並列計算(large scale synchronous poraelel computation)의 理論이며 거기에서는 逐次的 計算의 計算量 理論에서 領域量이 파라미터 S(n)으로 제약되는 것에 대해서 프로세서의 개수가 파라미터 H(n)(하드웨어의 H)으로 제약되는 것이다. 전형적으로 H(n)은 n에 관한 어떤 固定된 多項式이다.

여러 가지 逐次計算 모델이 있는(2. 참조) 것과 마찬가지로 매우 많은 並列計算 모델이 제안되어 있다(이들을 概觀하기 위해서는 文獻[21] 참조). 그러나 주되는 두개의 경쟁 모델이 있다. 첫번째 것은 共存메모리 모델의 클래스로 이 모델에서는 多數의 프로세서

⑦確率的 計算에 관한 더 많은 理論에 관해서는 Gill[32] 참조.

가 共存하고 있는 랜덤 억세스 메모리를 중계로 해서 서로 통신한다. 현실의 並列機械는 만들 때 이와 같은 형태로 만들어지는 것이 보통이므로 그 모델에 대해서 많은 並列 알고리즘이 발표되어 있다. 그러나 數學的 理論면에서는 그와같은 모델은 그다지 만족스러운 것이 되지 못한다. 그 이유는 그 詳細한 仕樣 指定에서 임의인 것이 너무 많기 때문이다. 共通메모리에 대한 읽기 쓰기의 경합은 어떻게 해결할 것인가? 각 프로세서에 대해서 어떤 基本的인 操作이 허용되는가? 共通메모리에 억세스하는데 log H(n) 時間單位의 코스트를 부과할 것인가?

따라서 나는 Borodin[8] (1977년)의 보다 깔끔한 모델쪽을 좋아한다. 그 모델에서는 並列計算機는 閉路를 갖지 않는(acgclic) 부울回路의 一樣한 族〈Bn〉이며 回路 Bn은 n 入力을 갖는다(따라서 길이 n인 入力系列만을 처리한다). 이때 H(n) (하드웨어量)은 단지 回路 Bn안의 게이트 數이고 T(n) (並列計算時間)은 回路Bn의 깊이(즉, 入力으로부터 出力에 이르는 最長經路의 길이)이다. 이 모델은(共有메모리 機械도 포함하여) 모든 현실적인 機械는 부울回路로 만들어진다는 것을 실제적으로 정당화한다. 또 어떤 函數를 計算하는데 필요한 부울回路의 最小 크기와 깊이는 자연스런 數學的 問題이며 並列計算의 理論이 나오기 이전에 이미 잘 考察되어 왔다.

並列計算理論에 있어서 다행한 것은 하드웨어量 H(n)이나 並列計算時間 T(n)의 最小값은 각종의 並列計算機 모델에 있어서 그다지 큰 차이가 나지 않는다. 특히 모든 모델에 대해서 點인 하나의 흥미있는 일반적 사실이 있다. 그것은 최초 Pratt와 Stockmeyer[58]에 의해서 1974년에 어떤 特定 모델에 대해서 증명되었고 文獻[33]에서는 "並列計算命題(parallel compatation thesis)"라 불린 것으로 그 사실은 다음과 같다. 즉 어떤 問題가(計算時間에 제한이 없는) 逐次的計算機械(Seguential machone, 順序機械)로 T(n)에 관한 多項式領域量으로 풀릴때 또 그때에 한에서 그 問題는 (하드웨어量에 제한이 없는) 並列計算機械로 T(n)에 관한 多項式時間에 풀린다.

並列計算에 있어서 기본적인 하나의 質問은 「어떤 問題가 1대의 프로세서로 하는 것보다 多數의 프로세서로 하는 쪽이 실질적으로 빨리 풀리는가?」이다. Nicholas Pippenger[59a]는 실제적으로 가능한 ($H(n)=n^{O(1)}$) 하드웨어量을 가지는 並列計算機로 超高速($T(n)=(\log n)^{O(1)}$ 時間)으로 풀리는 問題의 클래스(Nick의 클래스라는 의미로 현재 NC라 불리고 있다)를 정의하고 이 質問을 형식화하였다. 다행이도 클래스 NC는 택한 並列計算機의 특정 모델에 의해서 변하지는 않는다. 또 NC는 多項式時間에 逐次的으로 計算可能한 函數의 클래스 FP의 部分 클래스라는 것은 쉽게 알 수 있다. 따라서 우리의 非形式的 質問은 다음과 같이 形式化할 수 있다. 「FP에 속하는 어느 問題가 또 NC에도 속하는가?」

(그럴것 같지 않지만) NC=FP라고 상상해 볼 수도 있다. 왜냐하면 NC≠FP를 증명하려면 計算量理論에 있어서의 하나의 難關을 돌파해야 하기 때문이다(4.1의 마지막 참조). FP에 속하는 函數 F가 NC에 속하지 않는다는 것을 證明하는 방법을 우리는 모르고 있으므로 다음으로 할 수 있는 最善策은 f가 FP에 대해서 log 領域完全(log space complete)이라는 것을 증명하는 것이다. 이것은 어떤 문제가 NP完全임을 증명하는 것과 비슷한 생각이며 f에 대한 超高速並列알고리즘을 발견하려고 하는 노력에 대해서 용기를 잃게 한다는 실제적 효과를 갖는다. 그 이유는 f가 FP에 대해서 log 領域完全이고 또 f가 NC에 속하면 FP=NC라 結論 지을 수 있으며 한편 FP=NC가 성립하는 것은 놀랄만한 큰 사건이 되기 때문이다.

FP에 속하는 문제에 대해서 그것이 NC에 속하는가 여부, 또는 FP에 대해서 log 領域完全인가 여부(물론 그 어느쪽도 아닐 수도 있다)에 관해서 FP를 分類하는 연구에 많은 진전이 있었다. P에 대해서 完全한 문제의 최초의 예는 1793년에 文獻[20]에서 내가 제시하였다. 단지 나는 그것을 「完全性」에 관한 결과로서 기술하지는 않았지만…….

그보다 조금 후에 Jones와 Laaser[38]는 完全性의 概念을 정의하고 文脈自由文法의 空集合問題를 포함하는 약 5개의 문제 예를 들었다. FP에 대해서 完全이라고 증명되어 있는 가장 간단한 문제는 아마도 다음과 같은 소위 回路값 問題(circuit value problem)[49]일 것이다. 「부울回路와 그의 각 入力값이 주어지고 그 出力값을 구하라」. 나에게 가장 흥미있는 문제에는 Goldschlager, Shaw, Staples[34]에 의한 것으로, 각변에 (큰) 陽整數容量이 붙어있는 네트워크가 주어지고 그것을 통과하는 最大 흐름(의 奇偶)을 구하라는 문제이다. 흥미를 끄는 것은 그 完全性 證明의 巧妙함이다. 마지막으로 線型計劃問題는 FP에 대해서 完全이라고 하는 것에 대해서 언급하고자 한다. 이 문제에서 어려운 부분은 그 문제가 P에 속한다는 것(文獻[43] 참조)을 보이는 것이며 그뒤 完全性의 證明[26]은 바로 된다.

NC에 속하는 것이 알려진 문제중에는 다음과 같은 것이 있다. 2進數에 대한 4개의 算術演算(+, −, ×, ÷), 分類, 그래프 連結性判定 行列演算(곱셈, 逆, 行列式, 位數), 多項式의 最大公約多項式을 구하는 것, 文脈自由言語의 認識, 그래프의 最小被覆森(miniman spanning forest)을 구하는 것(文獻[11],[21],[63],[67b] 를 참조). 주어진 그래프의 最大 맷칭의 「크기」를 구하는 문제는 "랜덤(random)" NC(동전 던지기가 허용되는 NC)에 속한다는 것이 알려져 있다.[11] 그러나 실제로 最大 맷칭 자체를 구하는 문제가 랜덤 NC에 속하는가 여부는 흥미있는 未解決의 문제이다. 文獻[89],[67b]의 결과는 문제가 NC에 속한다는 것을 보이기 위한 일반적 방법을 제공하고 있다.

속하는 것으로 FP에 대해서 完全인가, 또는(랜덤) NC에 속하는가 여부가 알려져 있지않은 가장 흥미있는 문제는 두 整數의 最大公約數를 구하는 것이다. 그래프의 最大 맷칭이나 極大 크리크(maximal cligne)를 구하는 문제도 그렇지만 그외에도 또 分類하지 않으면 안될 흥미있는 문제가 많이 存在한다(文獻[88] 참조).

Ⅶ. 未 來

計算量理論의 분야는 광대한데 이 概說은 짧다는것을 다시한번 말하고자 한다. 내가 전혀 언급하지 않았거나 가볍게 언급한 主題에 대해서도 많은 분야가 있다. 이들 분야를 연구하는 분들께 사과를 드린다.

Yao[92]에 의해서 "計算的情報理論 (computational information theory)"라 불리었든 비교적 새롭고 흥미를 끄는 한 분야가 실제상 實行可能한 計算을 통해서 알 수 있는 情報라는 것을 생각하여 Shannon의 古典的 情報理論 위에 構築되었다. 이 主題는 公開鍵暗號系에 관한 Diffie와 Hellman[25]이나 Rivest, Shamin, Adleman[67a]의 論文에 크게 자극된 것인데 그의 「計算的」이라는 말의 根源은 計算의 理論을 사용해서 단일 有限長系列이 「랜덤」이라는 개념에 처음으로 의미를 부여한 Kolmogoroff[45]나 Chaitin[14a],[14b] 까지 거슬러 올라간다. Shamir[73] 및 Blum 과 Micali[7]에 의해서 고찰된 이 理論에 있어서의 하나의 흥미있는 생각은 과거의 비트를 사용해서 미래의 비트를 예측하는 것이 곤난할 것으로 생각되는 擬似랜덤系列(pseudorandom sequences)의 生成과 관련되어 있다. Yao[92]는 그와같은 系列의 존재는 確率的(多項式時間으로 풀리는) 클래스 R(5. 참조)이 어떤 決定性時間으로 풀리는 클래스에 속한다는 것을 의미한다는 것을 증명하였다. 사실 計算的情報理論은 計算에 있어서 랜덤性의 역할에 서광을 비칠 가능성이 있다.

計算的情報理論외에 確率的 알고리즘, 並列計算, (운이 좋으면) 下界에 관해서 흥미있는 새로운 결과가 나올 것을 기대할 수 있다. 下界에 대해서는 가까운 장래에 突破할 수 있을 것으로 내다보는데 있어서 하나의 難關은 P에 속하는 문제가 모두 0 (log n) 領域量으로 풀리지는 않는다는 것, 그리고 아마 또 P≠NC라는 것을 증명하는 것이다. 아무튼 計算量理論의 分野는 매우 活氣찬 연구가 계속될 것이다. 나는 未來가 가져다 줄 것을 즐거운 마음으로 지켜보고 있다.

謝 辭

많은 유익한 注釋과 示唆를 해준 Toronto大學의 計算量理論研究의 同僚, 특히 Allan Borodin, Joachim von zur Gathen, Silvio Micali 및 Charles Rackoff 에 감사한다.

參 考 文 獻

[1] Adleman. L.,"Two theorems on random polynomial time," *Proc. 19th IEEE Symp. on Foundations of Computer Science. IEEE Computer Society,* Los Angeles pp.75-83, 1978.

[2] Adleman. L., Pomerance, C., and Rumley R.S.,"On distinguishing prime numbers from composite numbers," *Annals of Math 117,* pp. 173-206, Jan. 1983.

[3] Aho, A.V., Hopcroft, J.E. and Ullman J.D., *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, Mass., 1974.

[4] Bennett, J.H. On *Spectra.,* Doctoral dissertation, Department of Mathematics, Princeton University, 1962.

[5] Berlekamp. E.R. *Factoring Polynomials Over Large Finite Fields.* Moth. Comp. 24 pp. 713-735, 1970.

[6] Blum, M.A, "A machine independent theory of the complexity of recursive functions," *JACM 14, 2,* pp. 322-336, Apr. 1967.

[7] Blum, M., and Micali S,"How to generate cryptographically strong sequences of pseudo random bit," *Proc. 23rd IEEE Symp. on Foundations of Computer*

Science. IEEE Computer Society, Los Angeles, pp. 112-117, 1982.

[8] Borodin, A., "On relating time and space to size and depth," *SIAM J.* Comp., pp. 733-744, 1977.

[9] Borodin, A.,"Structured vs. general models in computational complexity," In *Logic and Algorithmic* Monographic no. 30 de L 'Enseignement Mathematique Universite de Geneve, 1982.

[10] Borodin, A. and Cook S.A time-space tradeoff for sorting on a general sequential model of computatio, *SIAM J. Comput. 11* pp. 287-297, 1982.

[11] Borodin, A., von zur Gathen, J. and Hopcroft, J., Fast parallel matrix and GCD computations *23rd IEEE Sympon Foundations of Computer Science IEEE Computer Society, Los Angeles, 65-71, 1982.*

[12] Borodin, A. and Munro, I. *The Computational Complexity of Algebraic and Numeric Problems.* Elsevier, New York, 1975.

[13] Brockett, R.W. and Dobkin, D. "On the optimal evaluation of a set of bilinear forms," *Linear Algebra and its Applications 19,* pp. 207-235, 1978

[14a] Chaitin, G. J. On the length of programs for computing finite binary sequences, *JACM 13,* 4, 547-569, Oct. 1966. *JACM 16,* 145-159, Jan. 1969.

[14b] Chaitin, G. J. A theory of program size formally identical to informational theory, *JACM 22,* 3, pp. 329-340, July 1975.

[15] Cobham. A., The intrinsic computational difficulty of functions. *Proc. 1964 International Congress for Logic, Methology, and Philosophy of Sciences. Y.* Bar-Hellel, Ed., North Holland, Amsterdam, pp. 24-30, 1965.

[16] Cobham, A., The recognition problem for the set of perfect squares, *IEEE Conference Record Seventh SWAT.* pp. 78-87, 1966.

[17] Cohen, H. and Lenstra, H.W. *Jr. Primarily testing and Jacobi sums.* Report 82-18, University of Amsterdam, Dept. of Math., 1982.

[18[Cook, S.A., The complexity of theorem proving procedures. *Proc. 3rd ACM Symp. on Theory of Computing.* Shaker Heights, Ohio, pp. 151-158, May 3-5, 1971.

[19] Cook, S.A. Linear time simulation of deterministic two-way pushdown automata. *Proc. IFIP Congress 71,* (Theoretical Foundations), North Holland, Amsterdam, pp. 75-80, 1972.

[20] Cook, S.A. An observation on time-storage tradeoff. JCSS 9 (1974), 308-316 Originally in *Proc. 5th ACM Symp. on Theory of Computing,* Austin TX. pp. 29-33, Apr.30-May, 2 1973.

[21] Cook, S.A. Towards a complexity, theory of synchronous parallel computation. *L'Enseignement Mathematique XXVII* pp. 99-124, 1981.

[22] Cook, S.A. and Aanderaa, S.O. On the minimum computation time of functions. *Trans. AMS 142,* pp. 291-314, 1969.

[23] Cooley, J.M. and Tukey, J.W., An algorithm for the machine calculation of complex Fourier series, *Math Comput. 19* (1965), 297-301.

[24] Coppersmith, D. and Winograd, S. On the asymptomatic complexity of matrix multiplication, *SLAM J. Comp.* 11, pp. 472-429, 1982.

[25] Diffie, W. and Hellman, M.E. New directions in cryptography, *IEEE Trans. on Inform, Theory IT-226,* pp. 644-654, 1976.

[26] Dobkin, D., Lipton, R.J. and Reiss, S. Linear programming is log-space hard for *P. Inf. Processing Letters 8,* pp. 96-97, 1979.

[27] Edmonds, J. Paths, trees, flowers, *Canad. J. Math, 17,* pp. 449-67, 1965.

[28] Edmonds, J. Minimum partition of a matroid into independent subsets. *J.*

Res. Nat. Bur. Standards Sect. B, 69 pp. 67-72, 1965.

[29] Ferrante, J. and Rackoff, C.W., The Computational Complexity of Logical Theories. *Lecture Notes in Mathematics.* #718, Spiringer Verlag New York, 1979.

[30] Fischer, M. J. and Rabin, M. O. Super-exponential complexity of Presburger arikthmetic, In *Complexity of Computation, SLAM-AMS Proc. 7*, R. Karp. Ed., 974, 27-42.

[31] Garey, M.R. and Johnson, D.S. *Compputers and Introctability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, San Francisco, 1979.

[32] Gill, J., "Computational complexity of probabilistic Turing machines," *SLAM J. Comput. 6*, pp. 675-695, 1977.

[33] Goldschlager, L.M. Synchronous Parallel Computation. Doctoral dissertation. Dept. of Computer Sicence, Univ. of Toronto, 1977, See also *JACM 29 4*, pp. 1073-1066, Oct. 1982.

[34] Goldschlager, L. M., Shaw, R. A. and Staples, J. The maximum flow problem is log space complete for P. *Theoretical Computer Science 21*,pp.105-111, 1982.

[35] Grzegorczyk, A. Some classes of recursive functions, *Rozprawy Matematyczne*, 1953.

[36] Hartmanis, J. Observations about the development of theoretical computer science. *Annals Hist. Comput. 3, I*, pp. 42-51, Jan. 1981.

[37] Hartmanis, J. and Stearns, R.E. On the computational complexity of algorithms. *Trans. AMS 117*, pp. 285-306, 1965.

[38] Jones, N. D. and Laaser, W. T. Complete problems for deterministic polynomial time. *Theortical Computer Science 3*, pp.105-117, 1977.

[39] Kaltofen, E. A. polynomial reduction from multivariate to bivariate integer polynomial factorization. *Proc. 14th ACM Symp in Theory Comp.* San. Francisco, CA, pp.261-266, May 5-7 1982.

[40] Kaltofen, E. A polynomial-time reduction from bivariate to univariate integral polynomial factorization. *Proc. 23rd IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles* pp.57-64, 1982.

[41] Karatsuba, A. and Ofman. Yu., Multiplication of multidigit numbers on automata. *Doklady Akad. Nauk 145.2* (1962), 293-294. Translated in *Soviet Phys. Doklady, 77*, pp. 595-596, 1963

[42] Karp, R. M. Reducibility among combinatorial problems. In: *Complexity of Computer Computations.* R. E. Miller and J.W. Thatcher, Eds., Plenum Press, New York, pp.85-104, 1972.

[43] Khachian, L. G., A polynomial time algorithm for linear programming. *Doklody Akad. Nauk SSSR. 244, 5*,pp.1093-96, 1979. Translated in *Soviet Math. Doklady, 20,* 191-194.

[44] Knuth, D.E. *The Art of Computer Progromming.* vol. 3 Sorting and Searching, Addison-Wesley, Reading, MA. 1973.

[45] Kolmogorov. A. N., Three approaches to the concept of the amount of information. *Probl. Pered. Inf. (Prob), of Inf. Transm.)* 1 1965.

[46] Kolmogorov. A. N. and Uspenski, V. A., On the definition of an algorithm. *Uspehi Mat. Nauk. 13* pp 3-28, 1958. AMS Transl. 2nd ser. 29, pp 271-245, 1963.

[47] Ladner, R.E., The circuit value problem is log space complete for *P. SIGACT News 71* pp.18-20, 1975.

[48] Lenstra, A. K., Lenstra, H. W. and Lovasz L., Factoring polynomials with rational coefficients. *Report 82-05, University of Amsterdam. Dept. of Math.*, 1982.

[49] Levin. L. A., Universal search problems. *Problemy Peredaci Informacii 9* pp.115-116, 1973. Translated in *Problems of Information Transmission 9*, 265-266.

[50] Luks, E. M., Isomorphism of graphs of bounded valence can be tested in poly-

nomial time. *Proc. 21st IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles,* pp.42-49, 1980.

[51] Meyer, A.R. Weak monadic second-order theory of successor is not elementary-recursive. *Lecture Notes in Mathematics.* #453, Springer Verlag NewYork, pp.132 -154, 1975.

[52] Meyer, A. R. and Stockmeyer, L. J. The equivalence problem for regular expressions with squaring requires exponential space. *Proc. 13th IEEE Symp. on Switching and Automata Theory,* pp. 125 -129, 1972.

[53] Miller, G. L. Riemann's Hypothesis and tests for primality. *J. Comput. System Sci. 13* pp. 300-317, 1976.

[54] Oppen, D. C. A 2 upper bound on the complexity of Presburger arithmetic. *J. Comput. Syst. Sci. 16* pp.323-332,1978.

[55] Papadimitriou, C. H. and Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[56] Paterson, M.S., Fischer, M.J. and Meyer, A. R., An improved overlap argument for on-line multiplication, *SIAM-AMS Proc.* 7, Amer, Math. Soc., Providence, pp. 97-111, 1974

[57a] Pippenger, N. On simultaneous resource bounds (preliminary version). *Proc 20th IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles,* pp. 307-311, 1979.

[57b] Pippenger, N.J., and Fischer, M. J. Relations among compl, *JACM 26,* 2. pp. 361-381, Apr. 1979.

[58] Pratt, V. R. and Stockmeyer, L. J., A characterization of the power of vector machines. *J. Comput. System Sci. 12* (1976), 198-221. Originally in *Proc. 6th ACM Symp. on Theory of Computing Seattle,* WA, pp. 122-134, Apr. 30-May 2, 1974.

[59] Rabin, M. O., Speed of computation and classification of recursive sets. *Third Convention Sci. Soc., Israel,* pp. 1-2, 1959.

[60] *Rabin, M.* O., Degree of difficulty of computing a function and a partial ordering of recursive sets. Tech. Rep. No. 1, O.N.R., Jerusalem, 1960.

[61] Rabin, M. O., Probabilistic algorithms. In *Algorithms and Complexity, New Directions and Recent Trends,* J. F. Traub, Ed., Academic Press, New York pp. 21-39, 1976.

[62] Rabin, M. O., Complexity of Computations. Comm. *ACM 20, 9,* pp.625-633, 1977 1977.

[63] Reif, J. H., Symmetric Complementation *Proc. 14th ACM Symp. on. Theory of Computing,* San francisco, CA, pp. 201-214, May 5-7, 1982.

[64] Reisc, S. and Schnitger, G. Three applications of Kolmorgorov complexity. *Proc. 23rd IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles,* pp.45-52, 1982.

[65] Ritchie, R. W. *Closses of Recursive Functions of Predictable Complexity.* Doctoral Dissertation. Princeton University, 1960.

[66] Ritchie, R. W., Classes of predictably computable functions, *Trans. AMS 106,* pp.139-173, 1963.

[67a] Rivest, R. L., Shamir, A. and Adleman, L. A method for obtaining digital signatures and public-key cryptosystems, *ACM 21, 2,* pp.120-126, Feb. 1978.

[67b] Ruzzo, W. L., On uniform circuit complexity, *J. Comput. System Sci. 22* pp.365-383, 1981.

[68a] Savage, *J. E. The Complexity of Computing,* Wiley, New York, 1976.

[68b] Schnorr, C. P. The network complexity and the Turing machine complexity of finite functions. *Acta Informatical 7,* pp.95-107, 1976.

[69] Schonhage, A. Storage modification machines. *SLAM J. Comp. 9, 490-508, 1980.*

[70] Schonhage, A. and Strassen, V. Schnelle Multiplication grosser Zahlen. *Computing 7* pp.281-292, 1971.

[71] Schwartz, J. T. Probabilistic algorithms for verification of polynomial identities. *JACM 27, 4* pp.701-717, Oct. 1980.

[72] Schwartz, J. T. Ultracomputers. *ACM Trans. on Prog. Languages and Systems 2, 4* pp.484-521, Oct. 1980.

[73] Shamir, A. On the generation of cryptographically strong pseudo random sequences. 8th Int. Colloquium on Automata. Languages, and Programming Lecture Notes in Computer Science No. 115, Springer Verlag, New York, pp.544-550, July 1981.

[74] Shannon, C.E. The synthesis of two terminal switching circuits. *BSTJ 28* pp.59-98, 1949.

[75] Smale, S. On the average speed of the simplex method of linear programming. *Preprint,* 1982.

[76] Smale, S. The problem of the average speed of the simplex method. *Preprint,* 1982.

[77] Solovay, R. and Strassen, V. A fast monte-carlo test for primality. *SLAM J. Comput. 6* pp.84-85, 1977.

[78] Stearns, R. E., Hartmanis, J. and Lewis P. M. II. Hierarchies of memory limited computations. *6th IEEE Symp. on Switching Circuit Theory and Logical Design.* pp.179-190, 1965.

[79] Stockmeyer, L. J. The complexity of decision problems in automata theory and logic. Doctoral Thesis, Dept. of *Electrical Eng,* MTT, Cambridg, MA., Report TR-133, MTT Laboratory for Computer Science, 1974.

[80] Stockmeyer, L. J. Classifying the computational complexity of problems. Resrarch Report RC 7606, Math. *Science Dept.,* IBM T.J. Watson Resrarch Center, Yorktown Heights, N.Y. 1979.

[81] Strassen, V. Gaussian elimination is not optimal *Num, Math. 13* pp.354-356, 1969.

[82] Strassen, V. Die Berechnungskomplexital von elementarsymmetrischen. Funktionen und von Interpolationskoeffizienten. *Numer. Math.* 29 pp.238-251, 1973.

[83] Baur, W. and Strassen, V. The complexity of partial derivatives. *Preprint,* 1982.

[84] Toom, A. L. The complexity of a scheme of functional elements realizing the multiplication of integers. *Doklady Adad. Nauk. SSSR 150 3,* pp. 496-498, 1963. Translated in *Soviet Math. Doklady 3,* pp 714-716, 1963.

[85] Turing, A. M. On computable numbers with an application to the Entscheidnungsproblem. *Proc. London Math. Soc. ser. 2, 42,* pp 230-265, 1936-7. A correction biid. 43 pp 544-546, 1937.

[86] Valiant, L. G. The complexity of enumeration and reliability problems. *SIAM J' Comput. 8* pp.410-421, 1979.

[87] Valiant, L. G. The complexity of computing the permanent. *Theoretical Computer Science 8* pp.189-202, 1979.

[88] Valiant, L. G. Parallel Computation. *Proc. 7th IBM Japan Symp.* Academic 6 Scientific Programs. IBM Japan, Tokyo 1982

[89] Valiant, L. G., Skyum, S., Berkowitz, S. and Rackoff, C. Fast parallel computation on polynomials using few processors. *Preprint (Preliminary version in Springer Lecture Notes in Computer Science.* 118 pp.132-139, 1981

[90] von Neumann. J. A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contributions to the Theory of Games II.* H. W. Kahn and A.W. Tucker, Eds. Princeton Univ. Press, Princeton, NJ, 1953.

[91] Yamada. H. Real time computation and recursive functions not real-time computable. *IRE Transactions on Electronic Computers. EC-11* pp.753-

760, 1962.

[92] Yao, A. C. Theory and applications of trapdoor functions (Extended abstract). *Proc. 23rd IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles* pp. 80-91, 1982.

著 者 紹 介 **

1982年度 美國 計算機學會(ACM)의 튜링(Turing)賞은 1982年 10月 25日 Dallas에서 개최한 ACM總會에서 Toronto 大學의 計算機科學科 Stephen Arthur Cook 教授에게 수여되었다. 튜링賞은 計算機分野의 學術的 貢献에 대해서 ACM이 수여하는 최고의 상이다.

受賞 이유중에서 Cook 教授의 업적을 다음과 같이 들고 있다. "Cook 박사는 計算의 複雜度에 관한 우리의 理解를 획기적으로 진보시켰다. 計算理論에 관한 1971년 ACM SIGACT 심포지움에서 발표한 그의 論文「定理證明節次의 計算複雜度, The Complexity of Theorem Proving Precedures」는 NP 完全 理論의 기초를 확립하였다. NP 完全問題 클래스의 境界 및 그 性質에 관한 그 후의 研究는 과거 10년간, 計算機科學에 있어 가장 활발하고 중요한 研究活動의 하나였다. Cook 教授는 計算機科學의 기초분야에 큰 영향을 준 研究성과를 이룩한 것으로 잘 알려진 분이다. 그는 計算量理論, 計算에 있어서 時間-領域 트레이드 오프(trade off), 프로그래밍 言語의 論理에 관해서 위대한 貢献을 하였다. 그의 업적은 우아하고 통찰력이 풍부하다는 것이 특징이며 計算의 本質이 무엇인가라는 문제에 조명하고 있다."

1970년에서 1979년 사이 Cook 教授는 National Research Council로 부터 연구 조성비를 받아 광범한 연구를 수행하였다. 그는 또 1977~1978년도의 E. W. R. Staecie Memorial Fellowship의 受賞者였다. 많은 획기적인 논문의 저자인 그는 현재 NP 完全問題에 대한 "좋은" 알고리즘은 존재하지 않는다는 것을 증명하는 일에 몰두하고 있다.

ACM 튜링賞은 計算機科學의 진보에 위대한 貢献을 이룩한 영국의 教學者 .A. M. Turing을 기념하기 위하여 만들어진 것이다.