

國立中央大學

資訊工程研究所
碩士論文

通用型數域篩選因數分解法之參數探討

研究生：金家豪

指導教授：顏嵩銘博士

中華民國九十三年六月七日



國立中央大學圖書館 碩博士論文電子檔授權書

(93 年 5 月最新修正版)

本授權書所授權之論文全文電子檔，為本人於國立中央大學，撰寫之碩/博士學位論文。(以下請擇一勾選)

(☒)同意 (立即開放)

()同意 (一年後開放)，原因是：_____

()同意 (二年後開放)，原因是：_____

()不同意，原因是：_____

以非專屬、無償授權國立中央大學圖書館與國家圖書館，基於推動讀者間「資源共享、互惠合作」之理念，於回饋社會與學術研究之目的，得不限地域、時間與次數，以紙本、微縮、光碟及其它各種方法將上列論文收錄、重製、公開陳列、與發行，或再授權他人以各種方法重製與利用，並得將數位化之上列論文與論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

研究生簽名：_____

論文名稱：_____通用型數域篩選因數分解法之參數探討_____

指導教授姓名：_____顏 嵩 銘_____

系所：_____資訊工程_____所 ☐博士 ☒碩士班

學號：_____90532020_____

日期：民國 93 年 6 月 7 日

備註：

1. 本授權書請填寫並親筆簽名後，裝訂於各紙本論文封面後之次頁（全文電子檔內之授權書簽名，可用電腦打字代替）。
2. 請加印一份單張之授權書，填寫並親筆簽名後，於辦理離校時交圖書館（以統一代轉寄給國家圖書館）。
3. 讀者基於個人非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

國立中央大學碩士班研究生

論文指導教授推薦書

資訊工程 研究所 金家豪 研究生所提
之論文

(題 目)

通用型數域篩選因數分解法之參數探討

係由本人指導撰述，同意提付審查。

指導教授 顏尚銘 (簽章)

93 年 6 月 7 日

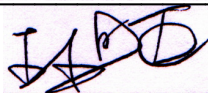
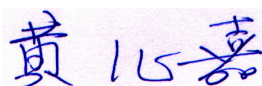
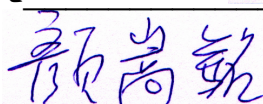
國立中央大學碩士班研究生
論文口試委員審定書

資訊工程 研究所 金家豪 君所提之論文
通用型數域篩選因數分解法之參數探討 經
(題 目)
本委員會審議，認定符合碩士資格標準。

學位考試委員會召集人



委 員



(簽章)

中 華 民 國 九 十 三 年 六 月 七 日

摘 要

論文名稱：通用型數域篩選因數分解法之參數探討

頁 數：58

校 所：國立中央大學 資訊工程研究所

研 究 生：金家豪

指導教授：顏嵩銘 教授

論文提要內容：

由於網際網路的興起，許多重要的資料開始經由網路傳輸，但如何保護傳輸資料不被他人竊取，便成為一個相當重要的議題。傳統的對稱式加密系統已不能滿足網際網路環境的需求，這使得公開金鑰密碼系統的需求日增。目前主流的公開金鑰密碼系統為 RSA，而 RSA 的理論安全性是建構在因數分解的難度上。因此，RSA 中的大數到底需要多大才算安全，自然就成為 RSA 系統安全的研究重點。到目前為止，分解 100 位數以上的 RSA 數最有效率的分解法為通用型數域篩選因式分解法。在本篇論文中，我們在一台 Linux Server 上實作該分解法，並藉由觀察參數調整的結果，以作為改進分解效能的建議。

關鍵字：整數分解、數域篩選因數分解法、通用型數域篩選因數分解法

Abstract

Title of Thesis: A study of parameter tuning for General Number Field Sieve

Pages: 58

Name of Institute: Department of Computer Science and Information Engineering,
National Central University

Name: Chia-Hao Chin

Advisor: Prof. Sung-Ming Yen

With the explosive growth rate of the Internet, there are more and more data transferred by Internet. Therefore, the ability to conduct electronic communications and transactions securely has become an issue of vital concern. The public key cryptosystem arises because the conventional secret-key cryptosystem can not meet the needs of distribution application. RSA is a very popular public key cryptosystem and its security relies on the difficulty of factoring a large number. Because of the popularity of this algorithm, much research has gone into this problem. As we know, the General Number Field Sieve(GNFS) is the asymptotically fastest factoring algorithm for large integers. In this thesis, we will implement this algorithm on a Linux server and discuss how to tune its parameters to get better performance.

Keyword: Factorization, Number Field Sieve, General Number Field Sieve

誌謝辭

本論文多蒙恩師 顏嵩銘教授細心的指導及教誨，才能得以順利完成，在此致上最誠摯的感謝。恩師不僅在學術上給予指導，在待人處世、治學態度上更令學生我受教良多。

口試論文時，承蒙口試委員：王旭正、孫宏民、黃心嘉教授對於論文的審查和內容各方面，給予諸多寶貴意見與指正，真是惠我良多，讓我不勝感激。特別是他們的學術觀點與獨特的分析，更令學生感到受益匪淺。

感謝中央數學系能讓我在擔任科技助教的工作之餘到資工系進修，並且要感謝數學系的助教們——坤展、晏森和至人，幫我解答許多數學上的問題。更要感謝中央資工所提供良好的學習與研究環境，讓我能一窺學術殿堂之美。

在此，我還要感謝我的老婆大人，感謝她長久以來對我的支持與鼓勵。最後，謹以一顆感恩的心，感謝所有曾經照顧過我、幫助過我的人，並願以此研究成果與他們共同分享。

金家豪 謹誌於

中央大學 資訊工程學系

中華民國九十三年六月

目 次

第 1 章 緒論.....	1
1.1 前言.....	1
1.2 研究動機.....	1
1.3 論文架構.....	3
第 2 章 RSA 密碼系統及相關數論介紹.....	4
2.1 RSA 密碼系統簡介	4
2.2 基礎數論.....	7
2.2.1 二次剩餘(Quadratic Residue)	7
2.2.2 雷建德符號(Legendre Symbol)	8
2.2.3 加寇比符號(Jacobi Symbol)	8
2.2.4 Smooth Number	9
第 3 章 因數相關分解演算法.....	10
3.1 試除法(Trial division)	10
3.2 費馬因數分解法(Fermat's factorization method)	11
3.3 Pollard's rho 因數分解法	14
3.4 Pollard's P-1 因數分解法	16
3.5 橢圓曲線因數分解法.....	17
第 4 章 因數無關分解演算法.....	20
4.1 Dixon 因數分解法(Dixon's factorization method).....	20
4.2 連分數因數分解法(Continued fraction factorization method).....	22
4.3 二次篩選因數分解法(Quadratic Sieve)	27
4.4 多多項式二次篩選因數分解法(Multiple Polynomial Quadratic Sieve)	33
4.5 數域篩選因數分解法(Number Field Sieve).....	37
第 5 章 通用型數域篩選因數分解法(General Number Field Sieve).....	38
5.1 選取多項式.....	39
5.2 篩選數對.....	40
5.2.1 選取有理數分解基底(Rational Factor Base)	41
5.2.2 選取代數數分解基底(Algebraic Factor Base)	41
5.2.3 選取二次特徵基底(Quadratic Character Base).....	42

5.2.4 檢驗數對.....	43
5.3 組成完全平方數.....	45
5.4 計算平方根.....	47
第 6 章 實驗結果與分析.....	51
6.1 實作環境.....	51
6.1.1 硬體環境.....	51
6.1.2 軟體環境.....	51
6.2 系統架構.....	51
6.3 實驗參數.....	53
6.4 實驗結果.....	53
6.5 數據分析.....	55
第 7 章 結論.....	56

圖目錄

圖 6-1 GNFS 的程式流程圖	52
圖 6-2 調整 a 的篩選範圍和 RFB 個數的數據比較圖	54

表目錄

表 6-1 調整篩選範圍和 RFB 的數據結果	54
-------------------------------	----

第1章 緒論

1.1 前言

傳統的密碼系統是以對稱金鑰密碼系統所建構出來，其方式為加密與解密時，均用相同一把金鑰作加解密動作，但如何能讓加密者與解密者雙方共享金鑰便成為這個系統必須克服的問題。拜網際網路的興盛，電子商務也迅速地蓬勃發展起來，許多重要的資料開始經由網路傳輸；同時，如何保護傳輸資料不被他人竊取，便成為一個相當重要的議題。由於傳統的對稱式密碼系統需要解決交換金鑰的問題，所以並不適合用於多人公開使用的網路環境，為了解決這個問題，Diffie 及 Hellman 於 1976 年[1,21]提出了公開金鑰 (public-key) 的概念，其概念為使用一對金鑰 (一個是公開給眾人的公開金鑰，另一個則為保持機密的私密金鑰)，由於利用公開金鑰所加密的文件只能由私密金鑰所解開，所以發文者可以利用收文者公開給眾人的公開金鑰對文件做加密，再把加密的文件送給收文者；收文者則利用只有他自己知道的私密金鑰對文件做解密，以得到發文者所送來的訊息。由於公開金鑰和私密金鑰有相關性，為了避免攻擊者由公開金鑰猜出私密金鑰，在設計公開金鑰密碼系統時，均利用數學上的難題特性，如：單向函數 (One-way Function) 或單向暗門函數 (One-way Trapdoor Function)，以確保系統的安全性。

1.2 研究動機

RSA 是當今主流的公開金鑰密碼系統，所以許多的系統安全都是建構在 RSA 的理論安全之上，而 RSA 的理論安全又是建構在因數分解的困難度上，所以，RSA 中的大數 N 到底需要多大以上才算安全，自然就成為研究重點。

因數分解問題是個古老的問題，而相關的研究也已經累積不少。一般而言，對於因數分解最常使用的演算法可以分成兩大類[2,3]：

一、因數分解所需花費的時間僅與欲分解的數 N 有關，跟它的因數沒有太大關係，這些方法有：

- (1) Lehman's method：這種方法的時間複雜度為 $O(N^{1/3+\epsilon})$ 。
- (2) Shanks' SQUARE FORM Factorization method (SQUFOF)：這種方法的

時間複雜度為 $O(N^{1/4})$ 。

- (3) Shanks' class group method：這種方法的時間複雜度為 $O(N^{1/5+\epsilon})$ 。
- (4) Continued FRAction (CFRAC) method：這種方法的時間複雜度為 $O(N^{c\sqrt{\log \log N / \log N}})$ ，其中 C 為常數，大約為 $\sqrt{2} \approx 1.4142$ 。
- (5) Multiple Polynomial Quadratic Sieve (MPQS)：這種方法的時間複雜度為 $O(N^{c\sqrt{\log \log N / \log N}})$ ，其中 C 為常數，大約為 $3/\sqrt{2} \approx 1.0606$ 。
- (6) Number Field Sieve (NFS)：這種方法的時間複雜度為 $O(N^{c\sqrt{\log N} \sqrt[3]{(\log \log N)^2}})$ ，由於 NFS 有兩種版本，所以 C 也會因所用的版本而不同：
 - 1. Special Number Field Sieve (SNFS)：是 NFS 針對型式為 $r^e \pm s$ 的整數所設計的，此時 C 的值大約為 $\sqrt[3]{32/9} \approx 1.526285657$ 。
 - 2. General Number Field Sieve (GNFS)：為 NFS 的一般型，適合用來分解一般的整數，此時 C 的值大約為 $\sqrt[3]{64/9} \approx 1.922999427$ 。

二、因數分解所需花費的時間與欲分解的因數 p 有關相當大的關係，這些方法有：

- (1) Trial division：這種方法的時間複雜度為 $O(p \times (\log(N)^2))$ 。
- (2) Pollard's ρ -method：這種方法的時間複雜度為 $O(p^{1/2} \times (\log(N)^2))$ 。
- (3) Lenstra's Elliptic Curve Method (ECM)：這種方法的時間複雜度為 $O(N^{c\sqrt{\log p \log \log p \times (\log N)^2}})$ ，其中 C 為常數，大約為 2。

以上兩大類的因數分解方法各有優缺點，第二大類的方法對於欲分解的數存在相當小的因數時十分有效率，但若分解的數之質因數都相當大時，其效率就明顯變差。而 RSA 中的大數是由強質數所組成，所以第二大類的因數分解法對於分解 RSA 的大數就不太適合。而在第一大類的因數分解法中，目前分解 100 位數以上的 RSA 數最有效率的分解法為通用型數域篩選因式分解法 (General Number Field Sieve，簡寫為 GNFS)，但這種演算法中，目前還有許多參數設定沒有很好標準可以確定其最佳值。本篇論文的内容為藉由實驗來觀察參數設定對其效能的影響，並建議該如何調整其參數以增加分解的效率。

1.3 論文架構

本論文於第二章會對 RSA 密碼系統作簡介，說明 RSA 是如何利用數論技巧進行加密、解密、簽章和驗章的動作，並分析因數分解對於 RSA 系統安全的重要性；並在第二章的後半部將介紹與因數分解相關的數學理論。

在第三、第四章會先回顧以往的因數分解法。在第三章中，介紹分解時間與欲分解數的因數有相當大關係的因數分解法，這類方法有：試除法、費馬因數分解法、Pollard's rho 因數分解法、Pollard's P-1 因數分解法以及橢圓曲線因數分解法；在第四章中，將介紹分解時間與欲分解數的因數沒有太大關係的因數分解法，這類方法有：連分數因數分解法 (Continued FRAction)、二次篩選因數分解法 (Quadratic Sieve)、多多項式二次篩選因數分解法 (Multiple Polynomial Quadratic Sieve) 以及數域篩選因數分解法 (Number Field Sieve)。

在第五章中，將介紹通用型數域篩選因數分解法 (General Number Field sieve) 的理論基礎；在第六章中，將實作通用型數域篩選因數分解法，並觀察藉由參數的調整對其效能的影響；並在第七章中，提出本篇論文值得改進的方向。

第2章 RSA 密碼系統及相關數論介紹

2.1 RSA 密碼系統簡介

RSA 密碼系統是在1978年由美國麻省理工學院三位教授 Rivest、Shamir 和 Adleman 所共同提出的公開金鑰密碼系統[1,21]，它是當今最常被使用的公開金鑰密碼系統，許多的系統安全性都是建構在 RSA 之上。RSA 是利用指數函數的可逆性和可交換性形成單向暗門函數，並利用因數分解的困難度來保護私密鑰匙不被攻擊者猜出。

在 RSA 密碼系統中，其公開金鑰包含一個相當大的合成數 N ，也就是 $N = p \times q$ ，其中 p 與 q 為大質數，以及一個小於 $\varphi(N) = (p-1) \times (q-1)$ 的正整數 e ，且 $\gcd(e, \varphi(N)) = 1$ ；而私密金鑰則為一個正整數 d ，其值等於模 $\varphi(N)$ 下 e 的反元素，亦即 $e \cdot d \equiv 1 \pmod{\varphi(N)}$ 。當發文者要對明文 m 作加密時，進行 $m^e \equiv c \pmod{N}$ 的運算，即可得到密文 c 。而當收文者要對 c 做解密時，則計算

$$c^d \equiv (m^e)^d \equiv m \pmod{N}$$

就可得到明文 m 。而若是要做數位簽章時，則簽章者將要簽署的明文 m 以自己的私密金鑰 d 作指數同餘運算，成為簽章 s ：

$$m^d \equiv s \pmod{N}$$

而需要對簽章的內容進行驗證簽章時，則檢查：

$$s^e \equiv (m^d)^e \equiv m \pmod{N}$$

是否成立。

若有攻擊者能很輕易地得知 N 是由那兩個質數所構成的話，那他就能知道 $\varphi(N)$ 的值為何，接下來他就能找出私密金鑰的 d 值，那這整個系統的安全性就被瓦解了。因此，RSA 密碼系統的安全性是建構在因數分解的難度上，所以 N 的值要多大，才能保障 RSA 密碼系統的安全，就成為一個重要的議題。

例 2.1：利用 RSA 進行秘密通訊

首先產生 RSA 金鑰對：

1. 選擇兩個質數 $p = 47$ 、 $q = 71$
2. 得到 $N = p \times q = 47 \times 71 = 3337$
3. 計算 $\phi(N) = (p - 1)(q - 1) = 46 \times 70 = 3220$
4. 選擇整數 $e = 79$ ，使得 $\gcd(\phi(N), e) = \gcd(3220, 79) = 1$ ，得到公開金鑰 $= \{e, N\} = \{79, 3337\}$
5. 計算 $d \equiv e^{-1} \equiv 79^{-1} \equiv 1019 \pmod{3220}$ ，得到私密金鑰 $= \{d, N\} = \{1019, 3337\}$

再來由發文方對文件作加密並傳送給收文方解密：

假設欲傳送的訊息為 $M = 688\ 232\ 687\ 966\ 668\ 003$ ，則發文方首先要得到收文方的公開金鑰，並利用該金鑰依序對 $m_1 = 688, m_2 = 232, m_3 = 687, m_4 = 966, m_5 = 668, m_6 = 003$ 在模 3337 下做 m_i^e 的運算

$$\begin{aligned}688^{79} \bmod 3337 &= 1570 = c_1 \\232^{79} \bmod 3337 &= 2756 = c_2 \\687^{79} \bmod 3337 &= 2091 = c_3 \\966^{79} \bmod 3337 &= 2276 = c_4 \\668^{79} \bmod 3337 &= 2423 = c_5 \\003^{79} \bmod 3337 &= 0158 = c_6\end{aligned}$$

所以加密後的密文 C 為：1570 2756 2091 2276 2423 0158；當收文方拿到加密的文件後，使用自己的私密金鑰對密文在模 3337 下做 c_i^d 的運算：

$$\begin{aligned}1570^{1019} \bmod 3337 &= 688 = m_1 \\2756^{1019} \bmod 3337 &= 232 = m_2 \\2091^{1019} \bmod 3337 &= 687 = m_3 \\2276^{1019} \bmod 3337 &= 966 = m_4 \\2423^{1019} \bmod 3337 &= 668 = m_5 \\0158^{1019} \bmod 3337 &= 003 = m_6\end{aligned}$$

最後得到原來的訊息 $M = 688\ 232\ 687\ 966\ 668\ 003$

例 2.2：利用 RSA 進行數位簽章

承例 2.1，若發文者要對傳送的訊息 $M = 688\ 232\ 687\ 966\ 668\ 003$ 進行簽章。首先，利用自己的私密金鑰依序對 $m_1 = 688, m_2 = 232, m_3 = 687, m_4 = 966, m_5 = 668, m_6 = 003$ 在模 3337 下做 m_i^d 的運算

$$688^{1019} \bmod 3337 = 2441 = s_1$$

$$232^{1019} \bmod 3337 = 1385 = s_2$$

$$687^{1019} \bmod 3337 = 1592 = s_3$$

$$966^{1019} \bmod 3337 = 0151 = s_4$$

$$668^{1019} \bmod 3337 = 2677 = s_5$$

$$003^{1019} \bmod 3337 = 2140 = s_6$$

會得到一個內容為 $S = 2441\ 1385\ 1592\ 0151\ 2677\ 2140$ 的簽章文件，再把文件內容 M 和 S 一起傳送給對方。

當對方收到文件 M 和 S 後，首先他取得發文者的公開金鑰，並對 $s_1 = 2441, s_2 = 1385, s_3 = 1592, s_4 = 0151, s_5 = 2677, s_6 = 2140$ 在模 3337 下做 s_i^e 的運算：

$$2441^{79} \bmod 3337 = 688 = m_1'$$

$$1385^{79} \bmod 3337 = 232 = m_2'$$

$$1592^{79} \bmod 3337 = 687 = m_3'$$

$$0151^{79} \bmod 3337 = 966 = m_4'$$

$$2677^{79} \bmod 3337 = 668 = m_5'$$

$$2140^{79} \bmod 3337 = 003 = m_6'$$

若收文者比對 m_i 和 m_i' 均相同，則表示所收到的文件沒有被更動；否則，就表示收到的文件和原始文件不同，如此就可確保文件的完整性。

2.2 基礎數論

2.2.1 二次剩餘 (Quadratic Residue)

定義 2.1：令 N 為正整數，若整數 a 滿足 $\gcd(a, N) = 1$ 且 $x^2 \equiv a \pmod{N}$ 有解，則 a 稱為 $\text{mod } N$ 之二次剩餘 (Quadratic Residue)，記為 QR_n ，否則 a 稱為 $\text{mod } N$ 之非二次剩餘 (Quadratic Non-Residue)，記為 QNR_n 。

例 2.3：若 $N = 7$ ，則 $\text{mod } N$ 之二次剩餘為 $\{1^2, 2^2, 3^2, 4^2, 5^2, 6^2\} \text{mod } 7 = \{1, 2, 4\}$ ，非二次剩餘則為 $\{3, 5, 6\}$ 。

定理 2.1：若 p 為奇質數 (>2) 且 $0 < a < p$ ，則

(1) 若 $a \in QR_p$ ，則 $x^2 \equiv a \pmod{p}$ 有二個解；若 $a \in QNR_p$ ，則

$x^2 \equiv a \pmod{p}$ 無解。

(2) $|QR_p| = \frac{p-1}{2}$ 且 $|QNR_p| = \frac{p-1}{2}$ 。

定理 2.2：若 p 為奇質數 (>2) 且 $0 < a < p$ ，則

$$a^{\frac{p-1}{2}} \pmod{p} = \begin{cases} 1 & \text{if } a \in QR_p \\ -1 & \text{if } a \in QNR_p \end{cases}$$

定理 2.3：令 p 為奇質數且 $c = a \times b \pmod{p}$ ，則

(1) 若 $a \in QR_p$ 且 $b \in QR_p$ ，則 $c \in QR_p$ 。

(2) 若 $a \in QR_p$ 且 $b \in QNR_p$ ，則 $c \in QNR_p$ 。

(3) 若 $a \in QNR_p$ 且 $b \in QR_p$ ，則 $c \in QNR_p$ 。

(4) 若 $a \in QNR_p$ 且 $b \in QNR_p$ ，則 $c \in QR_p$ 。

2.2.2 雷建德符號 (Legendre symbol)

若 p 為奇質數 (>2)，則雷建德符號 $\left(\frac{a}{p}\right)$ 定義為

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & \text{if } p \mid a \\ 1 & \text{if } a \in QR_p \\ -1 & \text{if } a \in QNR_p \end{cases}$$

2.2.3 加寇比符號 (Jacobi symbol)

若 $N = p_1^{e_1} \times p_2^{e_2} \times \cdots \times p_m^{e_m}$ 為奇正數，其中 p_1, p_2, \dots, p_m 為不同的質數，令 $\gcd(a, N) = 1$ ，則 Jacobi symbol 定義為

$$J\left(\frac{a}{N}\right) = J\left(\frac{a}{p_1^{e_1} \times p_2^{e_2} \times \cdots \times p_m^{e_m}}\right) = \left(\frac{a}{p_1}\right)^{e_1} \times \left(\frac{a}{p_2}\right)^{e_2} \times \cdots \times \left(\frac{a}{p_m}\right)^{e_m}$$

Jacobi symbol 的特性：

(1) 當 N 為奇質數時，則 $J\left(\frac{a}{N}\right) = \left(\frac{a}{N}\right)$ 。

(2) 當 N 為正奇數時，則 $J\left(\frac{1}{N}\right) = 1$ ， $J\left(\frac{-1}{N}\right) = (-1)^{\frac{N-1}{2}}$ 。

(3) 當 N 為正奇數時，則 $J\left(\frac{2}{N}\right) = (-1)^{\frac{N^2-1}{8}}$ ，亦即

若 $N \equiv 1 \pmod{8}$ 或 $N \equiv 7 \pmod{8}$ ，則 $J\left(\frac{2}{N}\right) = 1$ ；

若 $N \equiv 3 \pmod{8}$ 或 $N \equiv 5 \pmod{8}$ ，則 $J\left(\frac{2}{N}\right) = -1$ 。

(4) 當 a, b 為互質關係時，則 $J\left(\frac{a}{b}\right) \times J\left(\frac{b}{a}\right) = (-1)^{\frac{(a-1)(b-1)}{4}}$ 。

(5) 若 $N = p \times q$ ， p 與 q 為質數且 $x \in Z_n^*$ ，則 $J\left(\frac{x}{N}\right) = J\left(\frac{x}{p}\right) \times J\left(\frac{x}{q}\right)$ 。

其中 $Z_n^* = Z_n - \{0\}$ 。

2.2.4 Smooth Number

定義：如果某一個整數的所有質因數皆小於或等於 B 的話，則該整數可被稱之為 B -smooth 的數。

例如：

當 $B = 7$ 時， $2^{10} = 1024$ 是 B -smooth 的數；而 11 不是 B -smooth 的數。

第3章 因數相關分解演算法

根據算術基本定理 (Fundamental Theorem of Arithmetic)，任何一個比 1 大的正整數 N ，均可表示成

$$N = p_1^{e_1} \times p_2^{e_2} \times \cdots \times p_m^{e_m} \quad (3.1)$$

且此表示式為唯一。其中 p_1, p_2, \dots, p_m 均為質數且 $p_1 < p_2 < \dots < p_m$ ，而 e_1, e_2, \dots, e_m 均為正整數。如何為整數 N 找出除了 1 和本身之外的因數（這種因數稱之為 nontrivial factor）的問題，則稱之為因數分解問題 (Integer Factorization Problem，簡寫為 IFP)。在解決這個問題前，通常都會先找到和欲分解數 N 與其因數相關的方程式，再由這個方程式發展出其演算法。一般而言，依照與欲分解數 N 的因數之相關性，因數分解的演算法可以分成兩大類：第一大類的方法所需花費的時間與欲分解的因數 p 有關相當大的關係；而第二大類的方法所需花費的時間僅與欲分解的數 N 有關，與它的因數沒有太大關係。本章先闡述第一大類的方法，並在下一章闡述第二大類的方法。

3.1 試除法 (Trial Division)

若欲對於一個整數作因數分解，試除法[3,4]是最直觀的想法。若 N 是複合數，則在必存在一個大小介於 $1 \sim \sqrt{N}$ 的質因數。利用這個性質，只要拿 1 到 \sqrt{N} 的數去除 N ，若能整除，則表示找到 N 的因數。但這種方法在 N 相當大的時候，效率會變得很差，需要 \sqrt{N} 個除法來檢驗。最直覺的改進方法是想辦法減少檢驗的次數。由觀察發現，若從 1 開始依序去除 N ，對於測試數 q 而言，若 q 是複合數則不必測試，所以只需要用 1 到 \sqrt{N} 的質數做測試。根據質數定理 (Prime Number Theorem)，小於 x 的質數大約有 $x/\log x$ 個，因此只需做大約為 $\sqrt{N}/\log(\sqrt{N})$ 次的測試，就必定可以找出 N 的質因數。但是，當 N 很大時，我們要如何取得小於 \sqrt{N} 的質數就成了另一個問題。因此，一般在實作上，我們可以先定出一個上限值 C ，並利用質數測試法，把所有小於或等於 C 的質數找出來，並記錄在一個表中，如果我們要測試的數 N 滿足 $\sqrt{N} < C$ ，則能很快地利用上述的表中質數分解出 N 的因數；但當 N 大於 C 時，只好再利用質數測試的方法找出用來測試的質數，再進行檢驗。

3.2 費馬因數分解法 (Fermat's Factorization Method)

假設 N 是個合成數，並存在正奇數 p 和 q 且 $p \leq q$ 使得 $N = p \times q$ ，令 $x = \frac{1}{2}(p+q)$ ， $y = \frac{1}{2}(p-q)$ ，則 N 可以用以下的方式表示

$$N = x^2 - y^2 \quad (3.2)$$

如果能找到滿足 (3.2) 的一組 x, y ，就表示可以找到 N 的因數，這就是費馬因數分解法[3,4,5]主要的想法。那要如何找到這樣一組的 x, y 呢？首先，我們可以由 x 開始猜想其值為何，由於 x 必須大於 \sqrt{N} ，所以利用 $x_i = \lfloor \sqrt{N} \rfloor + i$ 猜測 x 值，其中 x_i 的下標代表這是第 i 次的猜測，若 $k_i = x_i^2 - N$ 是一個完全平方數，則表示我們找到正確的 x 值， y_i 值就等於 $\sqrt{x_i^2 - N}$ ，此時 p 值就等於 $x_i + y_i$ ， q 值就等於 $x_i - y_i$ ；若 k_i 不是完全平方數，我們就把 i 值加 1，再繼續測試。在計算 $k_i = x_i^2 - N$ 有個技巧可以避免計算 x_i^2 的值，其方法是

$$k_i = x_i^2 - N = (x_{i-1} + 1)^2 - N = x_{i-1}^2 + 2x_{i-1} + 1 - N = x_{i-1}^2 - N + 2x_{i-1} + 1 = k_{i-1} + 2x_{i-1} + 1$$

因此我們在計算 k_i 時，只需要計算 $k_{i-1} + 2x_{i-1} + 1$ ；而在檢查 k_i 是否為完全平方數的部份，由於完全平方整數的末兩位數字必定是下面 22 種情形之一 [22]：

00, 01, 04, 09, 16, 21, 24, 25, 29, 36, 41, 44, 49, 56, 61, 64, 69, 76, 81, 84, 89, 96

所以我們只需先檢查 k_i 的最後兩位數字是否為上述的 22 種情形之一。若不是，則可以肯定 k_i 不是完全平方數；若是，才需要檢查它是否為完全平方數。檢查是否為完全平方數的演算法如下：

輸入：正整數 a

輸出： a 是否為完全平方整數

(1) 把 a 模 100 後的結果給 b ；把 a 模 10 後的結果給 c ；把 b 除以 10 後的商結果給 d 。

(2) 檢查 b 是否為 0 或 25。若是，則跳到第 5 步；若不是，則繼續下

個步驟。

- (3) 檢查是否 c 為 1 或 4 或 9 且 d 為 0 或偶數。若是，則跳到第 5 步；若不是，則繼續下個步驟。
- (4) 檢查是否 c 為 6 且 d 為奇數。若是，則跳到第 5 步；若不是，則跳到第 7 步。
- (5) 把 a 開根號檢查結果是否為整數。若是，則跳到第 6 步；若不是，則跳到第 7 步。
- (6) 回傳 a 為完全平方整數。
- (7) 回傳 a 不為完全平方整數。

Fermat 因數分解法的演算法如下：

輸入：合成數 N 且 N 的質因數至少有一個以上

輸出：合成數 N 的質因數

- (1) x_0 設為 $\lfloor \sqrt{N} \rfloor + 1$ ，計算 $k_0 = x_0^2 - N$ 。
- (2) 測試 k_0 是否為完全平方數。若是，則跳到第 5 步，若不是，則把 i 設為 0。
- (3) 把 i 值加 1，並計算 $x_i = x_{i-1} + 1$ 和 $k_i = k_{i-1} + 2x_{i-1} + 1$
- (4) 測試 k_i 是否為完全平方數。若不是，則跳回第 3 步；若是，則繼續下個步驟。
- (5) 得到 N 的因數 $x_i - y_i$ 和 $x_i + y_i$ 。

例 3.1：利用 Fermat 因數分解法求出 2027651281 的因數

令 $N = 2027651281$. 則 $x_0 = \lfloor \sqrt{N} \rfloor + 1 = 45029 + 1 = 45030$

i	x_i	k_i	$2x_i + 1$
0	45030	49619	90061
1	45031	139680	90063
2	45032	229743	90065
3	45033	319808	90067
4	45034	409875	90069
5	45035	499944	90071
6	45036	590015	90073
7	45037	680088	90075
8	45038	770163	90077
9	45039	860240	90079
10	45040	950319	90081
11	45041	1040400	

當 $i=11$ 時 $k_i = 1040400 = 1020^2$ 代表找到滿足 (3.2) 的方程式，因此

$$2027651281 = (45041 - 1020) \times (45041 + 1020) = 44021 \times 46061$$

3.3 Pollard's Rho 因數分解法

Pollard's Rho 因數分解法[3,4,5,6]是在 1975 年由 John Pollard 所提出的因數分解演算法，對於分解的數若有很小的因數時，特別有效率。它的主要概念是想找尋是否存在一對 x 和 y 使得它們在模 d 時是相同的，但在模 N 是不同的，其中 d 是 N 的因數，則 $\gcd(x-y, N)$ 必定為 N 的因數。這可以用 (3.3) 來表示

$$\left. \begin{array}{l} x_i \equiv x_j \pmod{d} \\ x_i \not\equiv x_j \pmod{N} \end{array} \right\} \Leftrightarrow \left. \begin{array}{l} d \mid x_i - x_j \\ N \nmid x_i - x_j \end{array} \right\} \quad (3.3)$$

在一個循環序列中，很容易找到滿足第一個條件的數。我們可以利用 (3.4) 遞迴的形式造出一個循環序列，這樣子就有機會找出 N 的因數。

$$\left. \begin{array}{l} x_0 = \text{random}(0, N-1), \\ x_i \equiv f(x_{i-1}) \pmod{N} \end{array} \right\} \quad (3.4)$$

其中 x_0 是初始值， N 是要被分解的數，而函數 f 是一個係數為整數的多項式。依據這個遞迴公式，我們可以計算出一個由 x_0 開始的隨機序列 $S = x_0, x_1, x_2, \dots$

$$\left. \begin{array}{l} x_1 \equiv f(x_0) \pmod{N} \\ x_2 \equiv f(x_1) \equiv f(f(x_0)) \pmod{N} \\ x_3 \equiv f(x_2) \equiv f(f(x_1)) \equiv f(f(f(x_0))) \pmod{N} \\ \dots\dots\dots \\ x_i \equiv f(x_{i-1}) \pmod{N} \end{array} \right\} \quad (3.5)$$

由於序列 S 是有限且函數 f 是一個係數為整數的多項式，所以它必定形成一個循環的序列，因此必定會存在一組 x_i 和 x_j 使得 $x_i \equiv x_j \pmod{d}$ ，其中 $0 < i < j$ 。若 $k = j - i$ 且 $k > i$ ，則 $x_{2k} \equiv x_k \pmod{d}$ 。但我們無法事先得知 d 為何，所以無法計算 x_i ，不過由於 $(x - y) \pmod{d} = ((x - y) \pmod{N}) \pmod{d}$ ，所以我們可以藉由檢驗當 $i = 1, 2, 3, \dots$ 時，若 $\gcd(x_{2i} - x_i, N)$ 不等於 1，則表示我們找到 N 的因數。

Pollard's Rho 因數分解法的程序如下：

輸入：合成數 N 且 N 的質因數至少有一個以上

輸出：合成數 N 的質因數

- (1) 把 a, b 均設為 2 並把 i 設為 0。
- (2) 把 i 值加 1。把 a 值設為 $f(a) \bmod N$ ，把 b 值設為 $f(b) \bmod N$ ，再計算一次 $f(b) \bmod N$ ，並把結果給 b 。
- (3) 計算 $d = \gcd(a - b, N)$ 。若 $d = 1$ ，則跳回第 2 步；若 $1 < d < N$ 則表示找到 N 的因數，即可離開測試。

例 3.2：利用 Pollard's Rho 因數分解法找出 455459 的因數

令 $f(x) = x^2 + 1$ ，依照 Pollard's Rho 的演算法可以得到下列結果：

i	a	b	d
1	5	26	1
2	26	2871	1
3	677	179685	1
4	2871	155260	1
5	44380	416250	1
6	179685	43670	1
7	121634	164403	1
8	155260	247944	1
9	44567	68343	743

當 $i = 9$ 時， $d = 743$ ，所以 $455459 = 743 \times (455459/743) = 743 \times 613$

3.4 Pollard's P-1 因數分解法

根據 Fermat 定理，當 p 為一質數，且 $\gcd(a, p) = 1$ ，則 $a^{p-1} \equiv 1 \pmod{p}$ 。若存在整數 Q ，且 $p-1 \mid Q$ ，則 $p \mid a^Q - 1$ ，若我們檢查 $\gcd(a^Q - 1, N)$ 不等於 1 時，就表示找到 N 的一個因數，這就是 Pollard's P-1 因數分解法[3,4,5,7]的主要概念。但 Q 的值要如何決定？對於欲分解的數 N ，首先選定一個上限值 B ，求出比 $\lfloor \sqrt{N} \rfloor$ 小且為 B -smooth 的數之最小公倍數當作 Q 。若 $p-1$ 是比 $\lfloor \sqrt{N} \rfloor$ 小且為 B -smooth，則 $p-1$ 必能整除 Q ，如此就能利用計算 $\gcd(a^Q - 1, N)$ 找出 N 的因數。但若 Q 不是 $p-1$ 的倍數，則必須重新選擇上限值 B 值以獲得 Q 值，再進行計算。

Pollard's P-1 因數分解法的程序如下：

輸入：合成數 N 且 N 的質因數至少有一個以上

輸出：合成數 N 的質因數

- (1) 選定整數 B ，做為 B -smooth 的 B 值。
- (2) 從 2 到 $N-1$ 中選擇一個整數 a ，並計算 $d = \gcd(a, N)$ ，假如 $d > 1$ ，則回傳 N 的因數 d 。
- (3) 對於每一個小於 B 的質數 q

1. 計算 $l = \left\lceil \frac{\ln \lfloor \sqrt{N} \rfloor}{\ln q} \right\rceil$ 。

2. 計算 $a^{q^l} \bmod N$ ，並把結果給 a 。

- (4) 計算 $d = \gcd(a - 1, N)$ 。假如 $d > 1$ ，則回傳 N 的因數 d ；否則，就表示找不到 N 的因數，此時必須跳回到第 1 步重新選定 B 值。

例 3.3：利用 Pollard's P-1 因數分解法找出 19048567 的因數

選定 B 為 19；選擇 a 為 3，並測試 $d = \gcd(3, 19048567) = 1$

計算 q, l 的值的結果如下：

q	l	$a = a^{q^l} \bmod N$
2	12	12576225
3	7	14869166
5	5	3833788
7	4	10008585
11	3	13878469
13	3	11396399
17	2	1066763
19	2	4932455

計算 $d = \gcd(4932455-1, N) = \gcd(4932454, 19048567) = 5281$

因此 $N = 5281 \times (19048567/5281) = 5281 \times 3607$

3.5 橢圓曲線因數分解法

橢圓曲線分解法[3,8]是由 H. W. Lenstra Jr. 於 1985 年所提出的，主要的概念是利用橢圓曲線的特性來改善 Pollard's P-1 因數分解法的缺點。如上節所述，在 Pollard's P-1 因數分解法中，若 B 值選取不當導致無法分解成功時，唯有重選 B 值外別無他法。但若利用調整橢圓曲線的參數等於重新選擇不同曲線的性質，就可以避免這個問題。

首先介紹一個重要的性質，若在模 N 下無法找到 k 的反元素時，則

$\gcd(k, N)$ 必定為 N 的因數。而在橢圓曲線的運算過程中，經常會用到求反元素的計算。若無法進行反元素的運算時，則表示找到 N 的因數，這就是橢圓曲線因數分解法的主要技巧之一。

橢圓曲線的定義為所有滿足方程式 $E: y^2 = x^3 + ax + b$ ($x, y, a, b \in R$) 的點 (x, y) 所構成的集合。在模 N 下作運算，若方程式 $x^3 + ax + b$ 沒有重複的因式或是 $4a^3 + 27b^2 \neq 0$ ，則滿足方程式 $E: y^2 = x^3 + ax + b$ 的點 (x, y) 所構成的集合會形成一個群 (group)。在橢圓曲線中有一特殊的點 O ，但它並不在橢圓曲線上，稱為無限遠的點 (the point at infinity)。若橢圓曲線上點 P 的秩 (order) 為 n ，則表示 n 是最小的正整數使得 $nP = O$ 。

令橢圓曲線上有點 $P = (x_1, y_1)$ 和 $Q = (x_2, y_2)$ ，則其運算法則如下：

$$(1) \quad -O = O$$

$$(2) \quad -P = (x_1, -y_1)$$

$$(3) \quad P + O = O + P = P。$$

$$(4) \quad \text{若 } P = -Q, \text{ 則 } P + Q = O。$$

$$(5) \quad \text{若 } P \neq -Q, \text{ 則 } P + Q = (x_3, y_3), \text{ 其中}$$

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 \pmod{N} \\ y_3 &= \lambda(x_1 - x_3) - y_1 \pmod{N} \\ \lambda &= \frac{m_1}{m_2} \pmod{N} = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} \pmod{N} & \text{if } P \neq Q \\ \frac{3x_1^2 + a}{2y_1} \pmod{N} & \text{if } P = Q \end{cases} \end{aligned}$$

$$(6) \quad kP = \overbrace{P + P + \cdots + P}^{k-1 \text{ 次加法}}。$$

橢圓曲線因數分解法的演算法如下：

輸入：合成數 N 且 N 的質因數至少有一個以上

輸出：合成數 N 的因數

- (1) 選定 B 值，並求出 $k = B!$ 。
- (2) 選擇一條橢圓曲線 E 和在 E 上的點 P ，其中 E 上的每一點均屬於 $\mathbb{Z}/n\mathbb{Z}$ ，檢查 $d = \gcd(4a^3 + 27b^2, N)$ ，若 $d > 1$ ，表示找到 N 的一個因數，並跳到第 5 步；否則，繼續下一步。
- (3) 計算 kP ，若在計算過程中，若無法求出 m_2 的反元素時，則表示 $\gcd(m_2, N)$ 為 N 的一個因數，並跳到第 5 步；否則，繼續下一步。
- (4) 若 $kP = O$ ，則表示 $d = \gcd(m_2, N)$ 為 N 的一個因數；否則就表示找不到 N 的因數，這時就必須回到第 2 個步驟重新選取 E 或 P 。
- (5) 回傳 N 的因數。

例 3.4：利用橢圓曲線因數分解法找出 $N = 4453$ 的因數

- (1) 選定 B 值為 3，則 $k = 1 \times 2 \times 3 = 6$ 。
- (2) 選定橢圓曲線的參數 $a = 10$ ， $b = -2$ ，則 $E: y^2 = x^3 + 10x - 2$ ，並計算 $d = \gcd(4 \times 10^3 + 27 \times (-2)^2, 4453) = 1$ 。當 $x = 1$ 時，代入橢圓曲線 E 得到 $y = 3$ ，因此找到橢圓曲線 E 上的點 $P = (1, 3)$ 。
- (3) 計算 $kP = 6(1, 3)$ 。我們首先計算 $2P$ ，則必須先求出在 P 點時的斜率 $\lambda = \frac{3 \times x^2 + a}{2 \times y} \equiv \frac{3 \times 1^2 + 10}{2 \times 3} \equiv \frac{13}{6} \equiv 13 \times 3711 \equiv 3713 \pmod{4453}$ ，得到斜率後，就可計算

$$x_3 = \lambda^2 - x_1 - x_2 = 3713^2 - 1 - 1 \equiv 4332 \pmod{4453}$$

$$y_3 = \lambda(x_1 - x_3) - y_1 = 3713(1 - 4332) - 3 \equiv 3230 \pmod{4453}$$

因此 $2P = (4332, 3230)$ ，再來計算 $3P = P + 2P = (1, 3) + (4332, 3230)$ 。

首先，計算其斜率 $\lambda = \frac{m_1}{m_2} = \frac{y_2 - y_1}{x_2 - x_1} = \frac{3230 - 3}{4332 - 1} = \frac{3227}{4331}$ ，由於 $\gcd(m_2, N) =$

$\gcd(4331, 4453) = 61$ ，所以表示我們找到 4453 的因數 61，且 $4453 = 61 \times (4453/61) = 61 \times 73$ 。

第4章 因數無關分解演算法

4.1 Dixon 因數分解法

Dixon 因數分解法是 Dixon, J. D. 於 1981 年所提出來的[5,9]，這種分解法主要從 Fermat 因數分解改良出來的。假如存在一組滿足 (4.1) 式的 x, y 整數，則 $\gcd(x+y, N)$ 和 $\gcd(x-y, N)$ 必為 N 的因數。

$$\left. \begin{aligned} x^2 &\equiv y^2 \pmod{N} \\ x &\not\equiv \pm y \pmod{N} \end{aligned} \right\} \quad (4.1)$$

由於要找到同時滿足 (4.1) 中的兩個條件並不是那麼容易，所以退而求其次，先去找滿足第一個條件的一組數，但這還是不太容易，所以想辦法先得到滿足 (4.2) 的方程式

$$x^2 \equiv w \pmod{N} \quad (4.2)$$

再利用數個滿足 (4.2) 的方程式組合成滿足 (4.1) 第一個條件的方程式。但並不是每一個滿足 (4.2) 的方程式都會被拿來組合。首先，我們選定一個正整數稱之為 B ，若數值 w 的質因數均小於 B 的話，則稱 w 為 B -smooth。在篩選時，我們只留下 w 為 B -smooth 的方程式。若以 i 表示第幾個滿足 (4.2) 的方程式，則保留的方程式如(4.3)所示，而組合的方式可以用 (4.4) 來表示，其中 $a_i \in \{0,1\}$ ，當 $a_i=1$ 代表要選取第 i 次的結果組合；否則就不選取該次的結果。

$$\left. \begin{aligned} x_1^2 &\equiv w_1 = p_1^{e_{11}} p_2^{e_{21}} \cdots p_m^{e_{m1}} \pmod{N} \\ x_2^2 &\equiv w_2 = p_1^{e_{12}} p_2^{e_{22}} \cdots p_m^{e_{m2}} \pmod{N} \\ &\vdots \\ x_k^2 &\equiv w_k = p_1^{e_{1k}} p_2^{e_{2k}} \cdots p_m^{e_{mk}} \pmod{N} \end{aligned} \right\} \quad (4.3)$$

$$\prod_{i=1}^k a_i x_i^2 \equiv \prod_{i=1}^k a_i w_i \pmod{N} \quad (4.4)$$

但 a_i 的值要如何決定呢？由於 $\prod_{i=1}^k a_i x_i^2$ 必定是完全平方數，所以我們只需要使得 $\prod_{i=1}^k a_i w_i$ 是一個完全平方數，若它能分解成(4.5)，則它的質因數的指數必須是偶數，也就是(4.5)中的 e_1, e_2, \dots, e_m 必須均為偶數。

$$\prod_{i=1}^k a_i w_i = p_1^{e_1} p_2^{e_2} \cdots p_m^{e_m} \quad (4.5)$$

因此，上述問題可以改寫成為

$$\begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1k} \\ e_{21} & e_{22} & \cdots & e_{2k} \\ \vdots & & & \\ e_{m1} & e_{m2} & & e_{mk} \end{bmatrix} \times \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{bmatrix} \equiv \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \pmod{2} \quad (4.6)$$

若小於 B 的質數有 m 個，如果我們可以得到 $m+1$ 組以上滿足 (4.2) 的方程式，必定保證會有一組線性相依，這時則可利用高斯消去法取得 a_i 值後，再把得到的數個 (4.2) 方程式組合得到滿足 (4.5) 的方程式，若 $x \not\equiv \pm y \pmod{N}$ ，就可以得到 N 的因數為 $\gcd(x \pm y, N)$ 。

Dixon 因數分解法的演算法如下：

輸入：合成數 N 且 N 的質因數至少有一個以上

輸出：合成數 N 的質因數

(1) 選取 m 個分解基底 $\{p_1, p_2, \dots, p_m\}$ 。

(2) 任意選取數 x 。

(3) 計算 $x^2 \equiv w \pmod{N}$ ，若 w 能分解成 $p_1^{e_1} p_2^{e_2} \cdots p_m^{e_m}$ ，則保留分解結果，把 x 記做 x_i ， w 記做 $w_i = p_1^{e_{1i}} p_2^{e_{2i}} \cdots p_m^{e_{mi}}$ ，其中 i 代表保留的次序；否則，回到步驟 2。

(4) 若保留的分解結果個數不超過分解基底個數，則回到步驟 2；否則，

將保留結果的指數部份組成矩陣 $A = [e_{ij}]_{m \times (m+1)}$ ，並在模 2 下利用高斯消去法解 $A\beta = 0$ 。

(5) 依 β 的內容組合出 $x^2 \equiv y^2 \pmod{N}$ ，檢查 $\gcd(x \pm y, N)$ 是否為 N 的因數，若是，則回傳結果；若不是，則回到步驟 1，重新選擇分解基底；或回到步驟 2，重新選擇數 x 。

4.2 連分數因數分解法(Continued fraction factorization method)

連分數因數分解法[3,4,5,10]這個方法是由 M.A.Morison 和 J.Brillhart. 於 1975 年所提出來，主要的概念是利用連分數的表示法來滿足 (4.2) 方程式，再由一個以上的 (4.2) 方程式合併成滿足 (4.1) 的方程式組，進而達成因數分解 N 的目的。

對於任意開根號的正整數均可用 (4.7) 的表示法來表示，並將其稱之為其連分數的表示式

$$\sqrt{N} = A_0 + \frac{1}{A_1 + \frac{1}{A_2 + \frac{1}{\dots}}} = [A_0, A_1, A_2, \dots] \quad (4.7)$$

and $A_0, A_1, A_2, \dots \in \mathbb{N}$

例如：

$$\sqrt{2} = 1 + \frac{1}{\sqrt{2}-1} = 1 + \frac{1}{\sqrt{2}+1} = 1 + \frac{1}{1 + \frac{1}{\sqrt{2}+1} + 1} = 1 + \frac{1}{2 + \frac{1}{\sqrt{2}+1}} = 1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{\dots}}}$$

所以 $\sqrt{2}$ 可以表示成 $[1, \bar{2}]$ 。當我們找出 \sqrt{N} 的連分數表示式後，就可以用 (4.8) 做出 P_i 和 Q_i 的序列，並利用它去逼近 \sqrt{N} 。

$$\left. \begin{aligned} \sqrt{N} &= [A_0, A_1, A_2, \dots] \\ P_i &= A_i P_{i-1} + P_{i-2} \\ Q_i &= A_i Q_{i-1} + Q_{i-2} \\ P_{-2} &= 0, P_{-1} = 1, Q_{-2} = 1, Q_{-1} = 0 \end{aligned} \right\} \quad (4.8)$$

$$\text{則 } \left| \frac{P_i}{Q_i} - \sqrt{N} \right| \leq \frac{1}{Q_i^2}$$

例 4.1：利用 (4.8)，製作一個趨近於 $\sqrt{2}$ 的序列

i	-2	-1	0	1	2	3	4	5
A_i			1	2	2	2	2	2
P_i	0	1	1	3	7	17	41	99
Q_i	1	0	1	2	5	12	29	70

$$\{S_i\} = \left\{ \frac{P_i}{Q_i} \right\} = \left\{ \frac{1}{1}, \frac{3}{2}, \frac{7}{5}, \frac{17}{12}, \frac{41}{29}, \frac{99}{77}, \dots \right\}$$

當 i 很大的時候， $S_i \approx \sqrt{2}$ 。

利用方程式 (4.8)，我們可以求得序列 P_i 和 Q_i ，並作出

$$\left(\frac{P_i}{Q_i} \right)^2 - N = \frac{w_i}{Q_i^2} \quad (4.9)$$

而方程式 (4.9) 又可改寫成

$$P_i^2 - Q_i^2 N = w_i \quad (4.10)$$

方程式 (4.10) 與方程式 (4.2) 是等價的，就表示說我們可以利用 (4.10) 的關係式找出滿足 (4.2) 的方程式，若我們能得到 $m+1$ 條的 (4.2) 的方程式，就保證可以利用高斯消去法組合出一組以上的 x, y 值滿足 $x^2 \equiv y^2 \pmod{N}$ ，若 $x \not\equiv \pm y \pmod{N}$ ，則 $\gcd(x+y, N)$ 和 $\gcd(x-y, N)$ 必為 N 的因數。

連分數因數分解法的演算法如下：

輸入：合成數 N 且 N 的質因數至少有一個以上

輸出：合成數 N 的質因數

- (1) 計算 \sqrt{N} 的連分數表示式，得到 $\sqrt{N} = [A_0, A_1, A_2, \dots]$ 。
- (2) 決定 B 值，並作出 $FB = \{p_i \mid p_i < B\} \cup \{-1\}$ ，令小於 B 的質數個數有 m 個，並把 i 設為 -1 。
- (3) 把 i 的內容加 1，再利用 (4.8) 求得序列 P_i 和 Q_i ，並利用所得結果求出 $w_i = P_i^2 - Q_i^2 N \bmod N$ 的值。
- (4) 檢查 w_i 是否為 B -smooth。若是，則保留該方程式；若不是，則不保留。若保留方程式的個數超過 $m+2$ 個，則到下一步，否則回到第 3 個步驟。
- (5) 把第 4 步得到的方程式指數部份組成矩陣 A ，再把矩陣 A 模 2 得到矩陣 B ，並和單位矩陣 I 合併成擴增矩陣 C ，最後用高斯消去法找到相依行向量。
- (6) 由相依行向量可以得知，那些方程式可以組合出一組 x, y 值滿足 $x^2 \equiv y^2 \pmod{N}$ 。
- (7) 若 $x \not\equiv \pm y \pmod{N}$ ，則 $\gcd(x+y, N)$ 和 $\gcd(x-y, N)$ 必為 N 的因數。若 $x \equiv \pm y \pmod{N}$ ，則檢查另一個相依行向量。若所有相依行向量均檢查完時，還是找不到 N 的因數時，此時就必須跳到步驟 2 重新選取分解基底。

例 4.2：利用連分數因數分解法對 1037 做因數分解

- (1) 計算 \sqrt{N} 的連分數表示式，得到 $\sqrt{1037} = [32, \overline{4, 1, 15, 3, 3, 15, 1, 4, 64}]$
- (2) 選定 $B = 8$ ，則 $FB = \{-1, 2, 3, 5, 7\}$
- (3) 求出 A_i, P_i, Q_i, w_i

i	A_i	P_i	Q_i	w_i
0	32	32	1	-13
1	4	129	4	49
2	1	161	5	-4
3	15	2544	79	19
4	3	7793	242	-19
5	3	25923	805	4
6	15	396638	12317	-49
7	1	422561	13122	13
8	4	2086882	64805	-1
9	4	133983009	4160642	13
10	1	538018918	16707373	-49

- (4) 只有當 $i = 1, 2, 5, 6, 8, 10$ 時， w_i 為 B -smooth。因此，我們把這些方程式保留。

i	w_i	-1	2	3	5	7
1	49	0	0	0	0	2
2	-4	1	2	0	0	0
5	4	0	2	0	0	0
6	-49	1	0	0	0	2
8	-1	1	0	0	0	0
10	-49	1	0	0	0	2

- (5) 把步驟 3 得到的結果組成矩陣 A

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 2 \\ 1 & 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 \end{bmatrix}$$

再把 A 矩陣中的內容模 2 得到矩陣 B

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

把矩陣 B 和一個 6×6 的單位矩陣合併成擴增矩陣 C

$$C = \left[\begin{array}{cccccc|cccccc} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right]$$

再利用高斯消去法對 C 求相依行向量 β

$$C = \left[\begin{array}{cccccc|cccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array} \right]$$

得到 $\beta = [1 \ 0 \ 0 \ 0 \ 0 \ 0]^t$, $[0 \ 1 \ 0 \ 1 \ 0 \ 0]^t$, $[0 \ 0 \ 1 \ 0 \ 0 \ 0]^t$,
 $[0 \ 0 \ 0 \ 1 \ 1 \ 0]^t$, $[0 \ 0 \ 0 \ 0 \ 1 \ 1]^t$ 。

(6) 依 $\beta = [1 \ 0 \ 0 \ 0 \ 0 \ 0]^t$ ，所以我們選取 $i = 1$ 的方程式，由於

$\gcd(129 + \sqrt{49}, 1037) = 17$, $\gcd(129 - \sqrt{49}, 1037) = 61$ ，所以得到
 $1037 = 17 \times 61$ 。

4.3 二次篩選因數分解法 (Quadratic Sieve)

二次篩選因數分解法[3,4,5,11]是在 1982 年由 Carl Pomerance，所提出來的，由於連分數因數分解法花了太多的時間在測試 w_i 是否為 smooth，而二次篩選因數分解法利用埃拉托塞尼篩法 (Eratosthenes Sieve) 的概念來幫助檢測。首先，利用方程式 (4.11)

$$f(i) = w_i = \left(i + \left\lfloor \sqrt{N} \right\rfloor\right)^2 - N \equiv \left(i + \left\lfloor \sqrt{N} \right\rfloor\right)^2 \pmod{N} \quad (4.11)$$

做出符合(4.2)的方程式，並利用多項式的某些性質加速檢驗 $f(i)$ 是否為 B -smooth 的效率。

若 $f(i)$ 是 B -smooth，則 N 必是小於 B 的質數的二次剩餘，所以在選擇 FB 時，就可將小於 B 的質數中且 N 不是該質數的二次剩餘的數捨去。因此，

FB = $\{p_j : p_j \text{ is prime and } \left(\frac{N}{p_j}\right) = 1 \text{ for } p_j < B\}$ 。如此，就可縮小 FB 並加速

檢測 w_i 是否為 smooth 的效率。

由於

$$\begin{aligned} f(i + k \times p_j) &= \left((i + k \times p_j) + \left\lfloor \sqrt{N} \right\rfloor\right)^2 - N = \left((i + \left\lfloor \sqrt{N} \right\rfloor) + k \times p_j\right)^2 - N \\ &= \left((i + \left\lfloor \sqrt{N} \right\rfloor)\right)^2 + 2 \times \left((i + \left\lfloor \sqrt{N} \right\rfloor)\right) \times k \times p_j + (k \times p_j)^2 - N \\ &\equiv \left((i + \left\lfloor \sqrt{N} \right\rfloor)\right)^2 - N = f(i) \pmod{p_j} \end{aligned}$$

因此若 $p_j \mid f(i)$ ，則對於整數 k 會有 $p_j \mid f(i + k \times p_j)$ 的性質。利用這個性質，就可以利用埃拉托塞尼篩法來加速檢驗 $f(i)$ 是不是 B -smooth。

二次篩選因數分解法的演算法如下：

輸入：合成數 N 且 N 的質因數至少有一個以上

輸出：合成數 N 的質因數

令 $f(x) = x^2 \bmod N$

(1) 選定上界 B 值後，利用小於 B 的質數且 N 為這些數的二次剩餘集

合成 $FB = \{p_j : p_j \text{ is prime and } \left(\frac{N}{p_j}\right) = 1 \text{ for } p_j < B, j \in [1, m]\}$ 。

(2) 選定 M 值，把 x 值設為 $\lfloor \sqrt{N} \rfloor - M + 1$ ，再依序加 1，加到

$\lfloor \sqrt{N} \rfloor + M$ ，並代入 $f(x) = x^2 \bmod N$ 以得到 $2M$ 條方程式。

(3) 製作 $2M \times (\#FB + 4)$ 的表格，第一行表示這是第幾個方程式，第二行代表是 x 的值，第三行代表是 $f(x)$ 的值；第四行表示 $f(x)$ 的正負號，若 $f(x)$ 大於 0 時，則填 0，否則填 1；最後一行為 $r(x)$ ，代表除去 FB 內元素後的結果， $r(x)$ 的初始值為 $f(x)$ 的絕對值。第五行到倒數第二行表示 $f(x)$ 分解後的 p_j 冪次，其預設值為 0。令 j 表示 FB 中的第幾個元素，建立表格時， j 從 1 到 m 進行處理。處理行數為第 $(4+j)$ 行的方法為，首先在第 1 到第 p_j 列中找出 $r(i)$ 的倍數。若有找到，令其值為 s ，並將行數為 $(4+j)$ 且列數為 s 位置的內容加 1，並把 $r(s)$ 的值設為 $r(s)/p_j$ ；再來，將行數為 $(4+j)$ ，列數為 $(s+d \times p_j^k)$ 位置的內容加 1，並將 $r(s+d \times p_j^k)$ 的值設為 $r(s+d \times p_j^k)/p_j$ ，其中 d 由 1 開始遞增且 $d < 2M/p_j^k$ ， k 代表對 p_j 指數為 k 時的情況進行處理。若有找到，則對 k 值加 1，並再繼續檢查；若沒有找到，則 j 值加 1，檢查 FB 中的下一個元素。若處理完 FB 中所有元素後，若保留的行數超過 $\#FB+1$ 則進入下一個步驟，否則就必須擴大 M 值，以獲取足夠的方程式。

(4) 把 $r(x)$ 值為 1 的行保留，並把第 4 行到第 12 行的內容組成矩陣 A ，再把 A 矩陣中的內容模 2 並作轉秩得到矩陣 B ，這時就可以利用再利用高斯—喬丹消去法(Gauss-Jordan elimination method)對矩陣 B 處理，並把 pivot 調整至對角線的位置而得到矩陣 C 。在矩陣 C 行中，由於在沒有 pivot 的行中，元素為 1 且該列有 pivot 的列號

所相對應的方程式，是組成該行的相依方程式之一。所以，記下所有的相依方程式，就可以得到我們要求的相依行向量 β 。

- (5) 依據向量 β ，選取相對映的方程式，令 $x = \prod x_i$ ， $y = \sqrt{\prod f(x_i)}$ ， i 為選取的方程式，並檢查是否滿足 (4.1)。若滿足，則 $\gcd(x+y, N)$ 和 $\gcd(x-y, N)$ 必為 N 的因數，若找不到能滿足 (4.1) 的方程式，則必須回到步驟 2 重新選擇篩選範圍 M 或須回到步驟 1 重新選擇分解基底再進行分解。

例 4.3：利用二次篩選因數分解法對 12079 作因數分解

$$\text{令 } f(x) = x^2 \bmod N = x^2 \bmod 12079$$

- (1) 選定 B 值為 30，檢查小於 30 的質數中，12079 是否為其二次剩餘，其中 $\left(\frac{13}{12079}\right), \left(\frac{19}{12079}\right), \left(\frac{29}{12079}\right)$ 均不為 1，所以 $\text{FB} = \{2, 3, 5, 7, 11, 17, 23\}$ ， FB 的個數 $\#\text{FB} = 7$
- (2) 選定 M 值為 10，並計算 $f(x)$ 的值。由於 $\lfloor \sqrt{N} \rfloor = \lfloor \sqrt{12079} \rfloor = 109$ ，所以 x 值的範圍是從 $\lfloor \sqrt{N} \rfloor - M + 1 = 109 - 10 + 1 = 100$ 到 $\lfloor \sqrt{N} \rfloor + M = 109 + 10 = 119$ ，並代入 $f(x) = x^2 \bmod N$ 作計算。
- (3) 產生一個 20×12 的表格，第 1 行由 1 依序填到 20；第 2 行由 100 依序填到 119，再把計算所得到的 $f(x)$ 填到第 3 行，並把 $f(x)$ 值的絕對值填到第 12 行；由於第 1 列到第 10 列的 $f(x)$ 值均為負數，所以第 4 行的第 1 列到第 10 列的內容均為 1；第 11 列到第 20 列的 $f(x)$ 值均為正數，因此第 4 行的第 11 列到第 20 列內容均為 0；接下來處理第 5 行的內容，在第 1 到第 2 行中，只有 $r(101)$ 是 2 的倍數，所以把第 5 行中 $t = 2$ 的那列值加 1，並把 $r(101)$ 設為 $r(100)/2$ ；接著把第 5 行中列為 $t = 2, 4, \dots, 20$ 的內容加 1，並把 $r(103), r(105), \dots, r(119)$ 設為 $r(103)/2, r(105)/2, \dots, r(119)/2$ ，對於 $p = 2$ 指數為 1 的情況而言，如此檢查過程就算完成；接下來，針對 $p = 2$ 時指數為 2 的情況進行檢查，也就是檢查 $f(x)$ 中 $2^2 = 4$ 的倍數。首先在第 1 到第 4 行中找出 $r(x)$ 為 2 的倍數的列，由於找不到，所以對於 $p = 2$ 的檢查就算完成，這時就可

以檢查 $p = 3$ 的情形，依此方法填出第 6 行的內容。依照以上的步驟，繼續填寫其他行的值，並得到下列這張表格。

t	x	$f(x)$	-1	2	3	5	7	11	17	23	$r(x)$
1	100	-2079	1	0	3	0	1	1	0	0	1
2	101	-1878	1	1	1	0	0	0	0	0	313
3	102	-1675	1	0	0	2	0	0	0	0	67
4	103	-1470	1	1	1	1	2	0	0	0	1
5	104	-1263	1	0	1	0	0	0	0	0	421
6	105	-1054	1	1	0	0	0	0	1	0	31
7	106	-843	1	0	1	0	0	0	0	0	281
8	107	-630	1	1	2	1	1	0	0	0	1
9	108	-415	1	0	0	1	0	0	0	0	83
10	109	-198	1	1	2	0	0	1	0	0	1
11	110	21	0	0	1	0	1	0	0	0	1
12	111	242	0	1	0	0	0	2	0	0	1
13	112	465	0	0	1	1	0	0	0	0	31
14	113	690	0	1	1	1	0	0	0	1	1
15	114	917	0	0	0	0	1	0	0	0	131
16	115	1146	0	1	1	0	0	0	0	0	191
17	116	1377	0	0	4	0	0	0	1	0	1
18	117	1610	0	1	0	1	1	0	0	1	1
19	118	1845	0	0	2	1	0	0	0	0	41
20	119	2082	0	1	1	0	0	0	0	0	347

(4) 把 $r(x)$ 值為 1 的列保留，並把保留列的第 4 行到第 11 行的內容組成矩陣 A 。

$$A = \begin{bmatrix} 1 & 0 & 3 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 2 & 0 & 0 & 1 \\ 1 & 1 & 2 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 4 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

再把 A 矩陣中的內容模 2 並作轉秩得到矩陣 B

$$B = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

利用高斯－喬丹消去法(Gauss-Jordan elimination method)對矩陣 B 處理，並把 Pivot 調整至對角線的位置而得到矩陣 C

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

檢查沒有 pivot 的行，若該列元素為 1 且該列有 pivot，則相依行向量 β 的相對位置則記為 1，否則記為 0。例如：第 5 行中沒有 pivot，所以我們檢查

該行有 1 的列中是否有 pivot，發現第 2 和第 3 列滿足條件，所以相依行向量 β 為在第 2 和第 3 的元素填入 1 並由於該行行號為 5，所以在第 5 的位置也填入 1，其餘均填 0。因此，得到 $\beta = [0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0]^t$ ；依照相同的方法檢查第 6 行和第 9 行，可以得到 $\beta = [1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0]^t$ 和 $[0 \ 1 \ 1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1]^t$ 。

(5) 依照 $\beta = [1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0]^t$ ，所以我們可以利用符合條件的第 1, 2, 3, 4, 6 的方程式，也就是 $x = 100, 103, 107, 109, 111$ 的方程式組合。

$$f(100) = -2079 = -1 \times 3^3 \times 7 \times 11$$

$$f(103) = -1470 = -1 \times 2 \times 3 \times 5 \times 7^2$$

$$f(107) = -630 = -1 \times 2 \times 3^2 \times 5 \times 7$$

$$f(109) = -198 = -1 \times 2 \times 3^2 \times 11$$

$$f(111) = 242 = 2 \times 11^2$$

檢驗 $\sqrt{f(100) \times f(103) \times f(107) \times f(109) \times f(111)} \equiv 2175 \pmod{12079}$ 是否等於 $\pm 100 \times 103 \times 107 \times 109 \times 111 \equiv 9904 \pmod{12079}$ 。由於 $2175 \equiv -9904 \pmod{12079}$ ，因此，我們必須選擇其他組解，但另外二組解過檢驗經後，均不符合 $x \equiv \pm y \pmod{N}$ 要求。因此，必須回到步驟 2 重新選擇篩選範圍 M 或須回到步驟 1 重新選擇分解基底再進行分解。

4.4 多多項式二次篩選因數分解法 (Multiple Polynomial Quadratic Sieve)

多多項式二次篩選因數分解法 (Multiple Polynomial Quadratic Sieve, 簡稱為 MPQS) [4,12,13] 是由 R.D. Silverman 於 1987 年發展出來的。由於 Peter Montgomery 發現, 在二次篩選因數分解法中, 多項式的二次係數固定為 1, 當我們篩選的範圍為 $2M$ 時, $f(x)$ 的最大值約為 $2M\sqrt{N}$, 當 M 變大時, 就不容易找到符合 smooth 的數, 而多多項式二次篩選因數分解法藉由二次係數不必為 1, 使得 $f(x)$ 變小的方式, 藉以提高找到符合 smooth 數的機率。並且, 由於多項式不是固定的, 所以可以造出許多條多項式同時分解, 以減少分解所需要的時間, 這就是多多項式二次篩選因數分解法的主要概念。

首先令 $f(x) = ax^2 + bx + c$, 若 $b^2 \equiv 4ac \pmod{N}$ 且 $\left(\frac{a}{N}\right) = 1$, 則在模 N 下 $f(x)$ 會是一個完全平方數。在這裡我們所討論的 N 為一個奇數, 且不為質數, 所以 N 除 4 的餘數可能等於 1 或 3。但 $b^2 - 4ac$ 在除 4 的餘數只可能等於 0 或 1, 所以當 N 除 4 的餘數等於 3 時, 我們必須把 N 乘上一個倍數 k , 且 $k \equiv 3 \pmod{4}$ 。因此, 我們分解的對象便成了 kN , 所以我們可以得到 (4.12) 的關係式

$$b^2 - 4ac = kN \quad (4.12)$$

為了提高 $f(x)$ 在 $x \in [-M, M]$ 中符合 B -smooth 的機率, 我們希望把 $\sup |f(x)|$ 儘量變小, 也就是希望 $|f(-M)| \approx \left|f\left(-\frac{b}{2a}\right)\right| \approx |f(M)|$, 由這個關係式, 我們可以推測

$$\begin{aligned} |f(-M)| &\approx |f(M)| \\ \Rightarrow a(-M)^2 + b(-M) + c &\approx aM^2 + bM + c \\ \Rightarrow b &\approx 0 \end{aligned}$$

$$\begin{aligned}
& \left| f\left(-\frac{b}{2a}\right) \right| \approx |f(M)| \\
& \Rightarrow -\left(a\left(-\frac{b}{2a}\right)^2 + b\left(-\frac{b}{2a}\right) + c \right) \approx aM^2 + bM + c \\
& \Rightarrow \frac{b^2}{4a} - c \approx aM^2 + c \\
& \Rightarrow \frac{b^2 - 8ac}{4a} \approx aM^2 \Rightarrow \frac{2b^2 - 8ac}{4a} \approx aM^2 (\because b \approx 0) \\
& \Rightarrow \frac{2kN}{4a} \approx aM^2 \Rightarrow \frac{kN}{2M^2} \approx a^2 \\
& \Rightarrow a \approx \frac{\sqrt{2kN}}{2M} \\
& \Rightarrow c \approx \frac{-M\sqrt{kN}}{2\sqrt{2}}
\end{aligned}$$

令 $d^2 = a$ ，所以 $d \approx \sqrt{\frac{\sqrt{2kN}}{2M}}$ ，選擇質數 d 使得 $d \equiv 3 \pmod{4}$ 且

$\left(\frac{d}{kN}\right) = 1$ ，因為 $b^2 \equiv kN \pmod{4d^2}$ ，所以可以利用 Hegel's 定理找出 b 。令

$b_0 = (kN)^{(d+1)/4} \pmod{d}$ ，因為 d 是質數，所以 $\left(\frac{kN}{d}\right) \equiv (kN)^{(d-1)/2} \pmod{d}$ ，而

且 $kN \equiv 1 \pmod{4}$ ，因此

$$b_0^2 = (kN)^{(d+1)/2} = (kN)(kN)^{(d-1)/2} \equiv (kN)\left(\frac{kN}{d}\right) \equiv (kN)\left(\frac{d}{kN}\right) = (kN) \times 1 = (kN) \pmod{d}$$

而 $\gcd(2b_0, d) = 1$ ，所以在模 d 下 $2b_0$ 的反元素存在。

$$b_1 \equiv (2b_0)^{-1} \frac{kN - b_0^2}{d} \pmod{d}$$

求得 b_0 和 b_1 ，就可求出 $b = (b_0 + b_1 d) \pmod{a}$ 。並導出

$$b^2 = b_0^2 + 2b_0 b_1 d + b_1^2 d^2 \equiv kN \pmod{a}$$

由於 $kN \equiv 1 \pmod{4}$ ，所以 $b^2 \equiv kN \pmod{4a}$

得到 a 和 b 的值後，就可以求出 $c = \frac{b^2 - kN}{4a}$

求得 $f(x)$ 後，接下來的步驟就和二次篩選因數分解法的方法相當類似。

決定多多項式二次篩選因數分解法中的多項式係數的步驟

輸入：合成數 N 且 N 的質因數至少有一個以上

$f(x)$ 中 x 的最大值 M

輸出： $f(x)$ 中的係數 a, b, c

(1) 決定 k 值

若 $N \equiv 1 \pmod{4}$ ，則 $k = 1$ ；若 $N \equiv 3 \pmod{4}$ ，則 $k \equiv 3 \pmod{4}$

(2) 決定 a 和 d 的值

從 $\sqrt{\frac{\sqrt{2kN}}{2M}}$ 附近的數中，選取出滿足下列 3 個條件的數為 d 。

1. d 為質數

2. $d \equiv 3 \pmod{4}$

3. $\left(\frac{kN}{d}\right) = 1$

決定 d 值後， a 則為 d^2

(3) 算出 b 的值

1. 計算 $b_0 \equiv (kN)^{(d+1)/4} \pmod{d}$

2. 計算 $b_1 \equiv (2b_0)^{-1} \frac{kN - b_0^2}{d} \pmod{d}$

算出 b_0 和 b_1 ，就能算出 $b = b_0 + b_1 d \pmod{a}$ ，若計算出來的 b 為偶數，則 $b = b - a$ 。

(4) 算出 $c = \frac{b^2 - kN}{4a}$

例 4.4：當 $N=12079$ ，並選擇 x 的範圍為 $M=5$ ，求出多多項式二次篩選因數分解法的多項式係數。

(1) 決定 k 值

由於 $N \equiv 3 \pmod{4}$ ，在此選擇 $k=11$ ，所以 $kN=132869$ 。

(2) 決定 a 和 d 的值

$$\sqrt{\frac{\sqrt{2kN}}{2M}} = \sqrt{\frac{\sqrt{2 \times 132869}}{2 \times 5}} \approx 7.1798...，由於 7 為質數，且 $d=7 \equiv 3 \pmod{4}$ ，$$

$$\left(\frac{kN}{d}\right) = \left(\frac{11 \times 12079}{7}\right) = 1，所以選擇 d 值為 7，則 $a = d^2 = 49$ 。$$

(3) 算出 b 的值

$$1. \text{ 計算 } b_0 = (kN)^{(d+1)/4} = 132869^{(7+1)/2} \equiv 2^2 \equiv 4 \pmod{7}，因此 $b_0=4$ 。$$

$$2. \text{ 計算 } b_1 = (2b_0)^{-1} \frac{kN - b_0^2}{d} = (2 \times 4)^{-1} \frac{132869 - 4^2}{7} = 18979 \equiv 2 \pmod{7}，因此 $b_1 = 2$ 。$$

$b = b_0 + b_1 d = 4 + 2 \times 7 = 18 \pmod{49}$ 。由於計算出來的 b 為偶數，所以 $b = b - a = 18 - 49 = -31$ 。

(4) 算出 c 的值

$$c = \frac{b^2 - kN}{4a} = \frac{(-31)^2 - 132869}{4 \times 49} = -673，因此 $c = -673$ 。$$

因此，對於分解 12079 而言，多多項式二次篩選因數分解法的所使用的多項式可以為 $f(x) = 49x^2 - 31x - 673$ 。

4.5 數域篩選因數分解法(Number Field Sieve)

在前面三節中所介紹的三種方法，大體上都是想找到具有 (4.2) 的方程式，再利用數個 (4.2) 的方程式組合成滿足 (4.1) 的方程式。因此，這幾種方法都是先做出一個完全平方數，而多項式的最高幕次必須為 2。數域篩選因數分解法 [3,4] 最主要的改進有二點：第一，它不必先做出一個完全平方數，而是利用同態函數，只要等式的一邊組合出完全平方數，則等式的另一邊也會同時組出完全平方數；第二，它所使用的多項式最高幕次可以為 2 以上。

首先，對於欲分解的數 N 而言，若存在一個不可分解的多項式 $f(x)$ ，並且存在正整數 m 使得 $f(m) \equiv 0 \pmod{N}$ ，便可利用 $f(x)$ 的根 θ ，造出多項式環 $\mathbb{Z}[\theta]$ 。這時會存在一個 $\mathbb{Z}[\theta] \rightarrow \mathbb{Z}/N\mathbb{Z}$ 的同態函數 φ ，且 φ 具有下列性質：

$$\begin{aligned}\varphi(ab) &\equiv \varphi(a)\varphi(b) \pmod{N} \quad \forall a, b \in \mathbb{Z}[\theta] \\ \varphi(a+b) &\equiv \varphi(a) + \varphi(b) \pmod{N} \quad \forall a, b \in \mathbb{Z}[\theta] \\ \varphi(1) &\equiv 1 \pmod{N} \\ \varphi(\theta) &\equiv m \pmod{N}\end{aligned}$$

假設我們可以找到一個成員為 (a, b) 數對的集合 S ，並能滿足下列條件

$$\begin{aligned}\prod_{(a,b) \in S} (a+bm) &= y^2 \\ \prod_{(a,b) \in S} (a+b\theta) &= \beta^2\end{aligned}$$

令 $x = \varphi(\beta)$ ，則

$$x^2 \equiv \varphi(\beta)^2 \equiv \varphi(\beta^2) \equiv \varphi\left(\prod_{(a,b) \in S} (a+b\theta)\right) \equiv \prod_{(a,b) \in S} \varphi(a+b\theta) \equiv \prod_{(a,b) \in S} (a+bm) \equiv y^2 \pmod{N}$$

因此 $x^2 \equiv y^2 \pmod{N}$ ，這就滿足 (4.1) 的第一個條件，如此就有機會找到 N 的因數，這就是數域篩選因數分解法的主要核心概念。由於通用型數域篩選因數分解法是本篇論文的實作方法，所以將會在下一個章節詳細描述這個方法的實作細節。

第5章 通用型數域篩選因數分解法 (General Number Field Sieve)

數域篩選因數分解法 (Number Field Sieve) [13,14,15,16,17,18,19,20]是於 1988 年由 John Pollard 所提出的方法，並在 1990 年作為分解第九費馬數 (the ninth Fermat number) 的實作方法。針對不同的分解對象，數域篩選因數分解法有兩種形式：(一)特殊型數域篩選因數分解法 (Special Number Field Sieve，縮寫為 SNFS)，適用於 $n = c_1 r^t + c_2 s^u$ 形式的整數；(二)通用型數域篩選因數分解法 (General Number Field Sieve，縮寫為 GNFS)，適用於任意型式的整數。由於在這篇論文中，主要討論的是分解 RSA 中 N 的困難度，所以在本章會詳細討論通用型數域篩選因數分解法的實作方式。

通用型數域篩選因數分解法在實作上可以分成四個主要部份：

(1) 選取多項式

選定多項式的最高幕次，並決定多項式的係數。

(2) 篩選數對

決定分解所需要的基底，並找出滿足篩選條件的數對。

(3) 組成完全平方數

由滿足篩選條件的數對中，找出使得 $\prod_{(a,b) \in S} (a+bm)$ 和 $\prod_{(a,b) \in S} (a+b\theta)$

均為完全平方數的集合 S 。

(4) 計算平方根

找出整數 y ，使得 $\prod_{(a,b) \in S} (a+bm) = y^2$ ；找出 $\beta \in \mathbb{Z}[x]$ ，使得

$\prod_{(a,b) \in S} (a+b\theta) = \beta^2$ ，並計算 $x = \varphi(\beta)$ 。

以下分別對這四大步驟，詳細說明其實作細節。

5.1 選取多項式

令欲分解的數為 n 。首先，可以依據公式 $d \approx \sqrt[3]{3 \log(n) / \log(\log(n))}$ 決定多項式的最高冪次 d 為何。選取好 d 值之後，我們希望能造出不可分解 (irreducible) 的多項式 $f: \mathbb{Z} \rightarrow \mathbb{Z}$ ，其係數均為整數並且最高冪次項的係數為 1 (monic)，並存在一個整數 m 使得 $f(m) \equiv 0 \pmod{n}$ 。於是，我們令 $m \approx \lfloor n^{1/d} \rfloor$ ，再用 m 來表示欲分解的數 n ，因此 n 可以用 (5.1) 的方式表示。

$$n = m^d + a_{d-1}m^{d-1} + \cdots + a_1m + a_0 \quad (5.1)$$

利用 (5.1) 中得到的序列 $\{a_{d-1}, a_{d-2}, \dots, a_0\}$ ，做出 (5.2) 的多項式

$$f(x) = x^d + a_{d-1}x^{d-1} + \cdots + a_1x + a_0 \quad (5.2)$$

如此，就能滿足 $f(m) \equiv 0 \pmod{n}$ 的要求。檢驗 $f(x)$ 是否無法分解，若不是，則必須重新選擇 m ，再造出另一個多項式；若是，則為必存在 d 個 $f(x)$ 的根 $\theta \in \mathbb{C}$ ，並可定義 $\mathbb{Z}[\theta]$ 為

$$\mathbb{Z}[\theta] = \{x: x = a_{d-1}\theta^{d-1} + a_{d-2}\theta^{d-2} + \cdots + a_0\theta + a_0 \text{ for } \{a_j\} \subset \mathbb{Z}\} \quad (5.3)$$

且 $\mathbb{Z}[\theta]$ 會是一個多項式環，並存在一個同態函式 $\varphi: \mathbb{Z}[\theta] \rightarrow \mathbb{Z}/n\mathbb{Z}$ 滿足

$$\left. \begin{array}{l} (1) \varphi(ab) \equiv \varphi(a)\varphi(b) \pmod{n} \quad \forall a, b \in \mathbb{Z}[\theta] \\ (2) \varphi(a+b) \equiv \varphi(a) + \varphi(b) \pmod{n} \quad \forall a, b \in \mathbb{Z}[\theta] \\ (3) \varphi(1) \equiv 1 \pmod{n} \\ (4) \varphi(\theta) \equiv m \pmod{n} \end{array} \right\} \quad (5.4)$$

例 5.1：選取 GNFS 中所需要的多項式

以分解 $n = 45113$ 為例，首先選取 $d = 3$ ，則 $m \approx \lfloor n^{1/d} \rfloor = \lfloor 45113^{1/3} \rfloor = 35$ ，在此，我們選擇 $m = 31$ ，接下來就可以決定 $f(x)$ 。由於

$$45113 = 31^3 + 15 \cdot 31^2 + 29 \cdot 31 + 8$$

所以，多項式為

$$f(x) = x^3 + 15x^2 + 29x + 8$$

決定好 $f(x)$ 後，還必須檢查 $f(x)$ 在有理數中是否為不可分解，而檢查 $f(x)$ 在有理數中是否為不可分解的問題，其實就等同於 $f(x)=0$ 是否存在有理數 a 使得 $f(a)=0$ 的問題。由於 $f(x)=0$ 可能的有理數解為 $8/1$ 的因數，所以可能的解為 $\pm 1, \pm 2, \pm 4, \pm 8$ ，但由於 $f(x)$ 的係數均為正，我們只要檢查 $f(-1), f(-2), f(-4), f(-8)$ 是否為 0，由於 $f(-1)=-7, f(-2)=2, f(-4)=68, f(-8)=224$ 均不為 0，所以 $f(x)$ 在有理數中是不可分解的。因此，我們可以選取 $f(x) = x^3 + 15x^2 + 29x + 8$ 作為通用型數域篩選因數分解法中的多項式。

5.2 篩選數對

為了組合出完全平方數，因此我們必須對所得到的結果分解。並不是所有的得到結果我們都會拿來拼湊完全平方數，我們必須對所得到的結果進行篩選，所以我們會利用一個重要的概念：X-smooth 來決定如何選取結果。

定義 5.1：若一數能由某個 X 基底所組合成，則稱為該數是為 X-smooth 的數。

由於 GNFS 需要篩選出 $a+bm$ 在有理數分解基底（Rational Factor Base；縮寫成 RFB）中和 $a+b\theta$ 在代數數分解基底（Algebraic Factor Base；縮寫成 AFB）中均為 smooth 的數對 (a,b) ，因此需要二種分解基底。

5.2.1 選取有理數分解基底 (Rational Factor Base)

有理數分解基底的選取方式為：先決定一個上界值，把所有小於這個上界值的質數所形成的集合當作 RFB，也就是

$$\text{RFB} = \{p : p \text{ is prime and } p \leq M\} \text{ for some } M \in \mathbb{N} \quad (5.5)$$

例 5.2：選取有理數分解基底

若選定 RFB 的上界值為 29，則

$$\text{RFB} = \{2, 3, 5, 7, 11, 13, 17, 19, 23, 29\}$$

5.2.2 選取代數數分解基底 (Algebraic Factor Base)

若存在一個有限集合 $\{a+b\theta\} \subset \mathbb{Z}[\theta]$ ，其中 $a, b \in \mathbb{Z}$ ，而且對於任意的 $a+b\theta$ 而言，並不會存在一組 $\alpha, \beta \in \mathbb{Z}[\theta]$ ，使得 $\alpha\beta = a+b\theta$ 。因此，它在 $\mathbb{Z}[\theta]$ 的地位就類似於質數在整數的地位，因此就可以拿它來做為代數數的分解基底。但由於在電腦中不太容易處理 $a+b\theta$ ，所以我們必須藉由性質 5.1，把 $a+b\theta$ 轉換成 (r, p) 的形式。

性質 5.1：令 $f(x)$ 為係數為整數且最高冪次係數為 1 的不可分解多項式，而 $\theta \in \mathbb{C}$ 是 $f(x)$ 的根，則集合 $\{(r, p)\}$ 和集合 $\{a+b\theta\} \subset \mathbb{Z}[\theta]$ 會有一對一且映成的關係。其中 p 為質數， $r \in \mathbb{Z}/p\mathbb{Z}$ ， $f(r) \equiv 0 \pmod{p}$ ， $a+b\theta$ 是 first degree prime ideals。

因此，代數數分解基底的選取方式為：先決定一個上界值，再找出所有小於或等於這個上界值的質數且滿足性質 5.1 的數對 (r, p) ，做為 AFB 的成員。

例 5.3：選取代數數分解基底

若 AFB 中的 p 上界值為 103，則

$$\text{AFB} = \left\{ \begin{array}{l} (0, 2), (6, 7), (13, 17), (11, 23), (26, 29), (18, 31), (19, 41), (13, 43), (1, 53), \\ (46, 61), (2, 67), (6, 67), (44, 67), (50, 73), (23, 79), (47, 79), (73, 79), (28, 89), \\ (62, 89), (73, 89), (28, 97), (87, 101), (47, 103) \end{array} \right\}$$

5.2.3 選取二次特徵基底 (Quadratic Character Base)

要如何確定 $c+d\theta$ 在 $\mathbb{Z}[\theta]$ 是一個完全平方數，則可藉由性質 5.2 幫我們檢驗。

性質 5.2：令 S 是數對 (a, b) 的集合，且存在某些 $\alpha \in \mathbb{Q}(\theta)$ 使得

$$\prod_{(a,b) \in S} (a+b\theta) = \alpha^2 \quad \text{對於任意滿足性質 5.1 的數對 } (s, q) \text{ 而言，若}$$

$(s, q) \nmid \langle a+b\theta \rangle$ 且 $f'(s) \not\equiv 0 \pmod{q}$ ，則會有以下的性質

$$\prod_{(a,b) \in S} \left(\frac{a+bs}{q} \right) = 1$$

因此，我們可以用比 AFB 中最大的 p 再大一點的質數為起點，開始找出滿足性質 5.1 的數對，以構成二次特徵基底 (Quadratic Character Base；縮寫成 QCB)

例 5.4：選取代數數分解基底

承接範例 5.3，由於 AFB 中 p 的最大數為 103，所以我們由 $p = 107$ 開始找出滿足性質 5.1 的數對，若 QCB 中 p 的上界值為 109，則

$$\text{QCB} = \{(4,107), (8,107), (80,107), (52,109), (99,109)\}$$

5.2.4 檢驗數對

如何確認 $a+b\theta$ 是否為 smooth，首先要利用性質 5.3 幫助我們確認 $c+d\theta$ 是否可以整除 $a+b\theta$ 。

性質 5.3: 對於在 $\mathbb{Z}[\theta]$ 中的元素 $c+d\theta$ 而言, $c+d\theta$ 可以整除 $a+b\theta \in \mathbb{Z}[\theta]$ 若且為若 $a \equiv -br \pmod{p}$, 其中 a 和 b 互質, 而 (r, p) 是 $c+d\theta$ 利用性質 5.1 得到的數對。

並且利用性質 5.4 幫助我們確定是否找到 $a+b\theta$ 的分解式。

性質 5.4: 有限集合 U , 其元素 $(r, p) \in \mathbb{Z}[\theta]$, 可以構成 $a+b\theta \in \mathbb{Z}[\theta]$ 的分解式 若且為若 $\prod_{(r_i, p_i) \in U} p_i = (-b)^d f(-a/b)$, 其中 d 為 $f(x)$ 的次數 (degree)。

對於固定的 b 和 m 而言, 要找出介於 $-M$ 到 M 中, 且 $a+bm$ 為 p 的倍數的整數 a , 可以利用下列這個性質: 對於任意整數 k 而言, 若 p 整除 $a+bm$, 則 p 必整除 $(a+pk)+bm$, 若找到符合條件的最小值 a , 利用上述技巧, 我們可以很快地篩選出 p 的因數。

但要如何找到符合條件的最小值 a 呢? 首先, 我們知道 $-p \times \lfloor M/p \rfloor$ 是所有比 $-M$ 大且為 p 倍數的整數中最小整數, 所以我們可以拿 $-p \times \lfloor M/p \rfloor$ 為 a 的初始值, 但 bm 不一定是 p 的倍數, 所以我們必須把 $-p \times \lfloor M/p \rfloor$ 再減去 $b \times (m \bmod p)$, 為了確保 $-p \times \lfloor M/p \rfloor - b \times (m \bmod p)$ 會大於 $-M$ 並且為 p 的倍數, 因此要找最小的正整數 h 使得 $-p \times \lfloor M/p \rfloor - b \times (m \bmod p) + ph > -M$ 。所以, 我們可以令 a 的初始值為 $-p \times \lfloor M/p \rfloor - b \times (m \bmod p) + ph$ 。而在檢驗 $(-b)^d f(-a/b)$ 是否為 smooth 時, 需要找出滿足 $a+br$ 會被 p 整除的數, 所以也可以利用上述技巧進行檢驗。

若 $a+bm = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ ，則 $\ln(a+bm) = e_1 \ln(p_1) + e_2 \ln(p_2) + \cdots + e_k \ln(p_k)$ ，由於 $\ln(p_i), p_i \in \text{RFB}$ 可以事先計算，所以檢測的動作，就可由需要較大運算量的除法改成較小運算量的加法，若 $a+bm$ 為 smooth，則 $\ln(a+bm)$ 減去 $e_1 \ln(p_1) + e_2 \ln(p_2) + \cdots + e_k \ln(p_k)$ 的結果就應該趨近於 0，運用這種技巧，可以增加篩選的速度。

由於在 RFB 中的檢測對象為 $a+bm$ ，所以我們定義 $\text{Norm-RFB}(a,b)$ 為 $a+bm$ ；而在 AFB 中的檢測對象為 $(-b)^d f\left(-\frac{a}{b}\right)$ ，因此定義 $\text{Norm-AFB}(a,b)$ 為 $(-b)^d f\left(-\frac{a}{b}\right)$ ，針對我們要檢測對象，利用上述的技巧檢測其是否為 smooth。

我們可以利用下面的步驟找到一些在 \mathbb{Z} 和 $\mathbb{Z}[\theta]$ 均為 smooth 的數對。

1. 首先選取 b 為 1。並開始檢驗 $a+bm$ 是不是 smooth。
2. 建立一個陣列，其內容為 $\ln(|a+bm|)$ ，其中 a 的範圍從 $-M$ 到 M ，而 M 是 a 的篩選範圍

$$\begin{bmatrix} \ln(|-M+bm|) \\ \ln(|(-M+1)+bm|) \\ \vdots \\ \ln(|(M-1)+bm|) \\ \ln(|M+bm|) \end{bmatrix}$$

3. 令 $\text{RFB} = \{p_1, p_2, \dots, p_k\}$ 。首先我們處理當質數為 p_1^e 的情形，其中 e 從 1 開始遞增的整數。利用 $-p_1^e \times \left\lfloor M / p_1^e \right\rfloor - b \times (m \bmod p_1^e) + p_1^e h + M$ ，找到第一個有 p_1^e 因數的位置 s ，並把該位置的內容減 $\ln(p_1)$ ，再去找從該位置 s 加上 $p_1^e k$ 的位置，其中 k 值為 1 開始遞增的正整數，並把該位置的值減 $\ln(p_1)$ ，其中 $s + p_1^e k < 2M$ 。利用這種方式， e 由 1 開始篩選，直到在陣列中找不到 p_1^e 的因數為止。

4. 利用步驟 3 的方法，依次處理完 p_1, p_2, \dots, p_k 的情形後，此時矩陣內容為 0 值的位置，代表該數必定是 smooth，就可把該值保留；否則，就註記該位置不是 smooth。
5. 檢驗 $(-b)^d f\left(-\frac{a}{b}\right)$ 是不是 smooth。首先，利用步驟 4 所得到的陣列做初始化，若該位置註記不是 smooth，則不必處理；否則，將其值設成 $\ln\left((-b)^d f\left(-\frac{a}{b}\right)\right)$
6. 若 $\text{AFB} = \{(r_1, p_1), (r_2, p_2), \dots, (r_k, p_k)\}$ ，首先我們處理質數為 (r_1, p_1) 的情形。利用 $-p_1 \times \lfloor M / p_1 \rfloor - b \times (r \bmod p_1) + p_1 h + M$ ，找到第一個有 p_1 因數的位置 s ，並把該位置的值減 $\ln(p_1)$ ，再去把該位置 $s + p_1 k$ 的內容減 $\ln(p_1)$ ，其中 $s + p_1 k < 2M$ 。
7. 利用步驟 3 的方法，依次處理完 $(r_1, p_1), (r_2, p_2), \dots, (r_k, p_k)$ 的情形後，此時矩陣內容為 0 值的位置，必定是在 RFB 和 AFB 均為 smooth 的數對。就可把該值選取。
8. 這時 $b = 1$ 時已檢驗完成，若選取的數對沒有到達所需的數量，則我們可以不斷地增加 b 值以尋找滿足在 RFB 和 AFB 均為 smooth 的數對，直到達到要求的數量為止。

5.3 組成完全平方數

在這一節中，將利用高斯消去法找出可以組成完全平方數的數對。若 RFB 的元素個數有 t 個，AFB 的元素個數有 u 個，QCB 的元素個數有 v 個，首先要造出一個 $(t+u+v+1) \times (t+u+v+2)$ 的矩陣，矩陣的每一行是由一對 (a, b) 所構成，而第一列為代表 $a + bm$ 的正負號，若為正值，則填 0；若為負值，則填 1。接下來的 t 列代表 $a + bm$ 分解質數的指數再模 2 的結果，也就是若 $a + bm = p_1^{e_1} p_2^{e_2} \cdots p_t^{e_t}$ ，則要填入的內容為 $e_1 \bmod 2, e_2 \bmod 2, \dots, e_t \bmod 2$ 。再接下來的 u 列代表 $a + b\theta$ 被 AFB 分解的結果，若 $a + b\theta$ 是其倍數，則在其相對位置填入 1；否則填入 0。最後的 v 列則填入計算 $\left(\frac{a + bs}{q}\right)$ 的結果，其

中 (s, q) 是 QCB 的元素，若 $\left(\frac{a+bs}{q}\right) = 1$ 則填 0；否則填 1。把所有的數對依照上述方法進行計算，並將所得結果填入矩陣中，再利用高斯－喬丹消去法在模 2 下對矩陣計算出一組線性相依的解，這個解就是我們所要的完全平方數。

例 5.5：

以 $(-8, 3)$ 為範例，建立相對於該行的矩陣內容。

承範例 5.2 中的 RFB，若 $(-8, 3)$ 在 RFB 為 smooth，則

$$a + bm = -8 + 3 \cdot 31 = 85 = 2^0 \cdot 3^0 \cdot 5^1 \cdot 7^0 \cdot 11^0 \cdot 13^0 \cdot 17^1 \cdot 19^0 \cdot 23^0 \cdot 29^0$$

因此，對應於 RFB 而言，其內容為 $(0, 0, 1, 0, 0, 0, 1, 0, 0, 0)$

承範例 5.3 中的 AFB，計算 $(-b)^d f\left(-\frac{a}{b}\right) = a^3 - 15a^2b + 29ab^2 - 8b^3 = -5696$ ，

而 $-5696 = -1 \cdot 2^6 \cdot 89^1$ ，但在 AFB 中，當 $p = 89$ 時，有三組 r 使得 $f(r) \equiv 0 \pmod{p}$ ，其分別為 $(28, 89)$ ， $(62, 89)$ ， $(73, 89)$ ，我們可以利用性質 5.3 確定那組 (r, p) 可以整除 $-8 + 3\theta$ ，由於 $-8 \equiv 81 = -3 \cdot 62 \pmod{89}$ ，所以代表 $c + d\theta \in \mathbb{Z}[\theta]$ 的 $(62, 89)$ 可以整除 $-8 + 3\theta$ 。因此，對應於 AFB 而言，其 23 個 bit 內容應為 $(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0)$

承範例 5.4，把 QCB 中的元素代入 $\left(\frac{a+bs}{q}\right)$ 計算結果。例如，當 $(s, q) = (80, 107)$ 時， $\left(\frac{-8+3 \cdot 80}{107}\right) = -1$ ，因此在代表 $(80, 107)$ 的位置應填入 1。把全部 QCB 元素代入 $\left(\frac{a+bs}{q}\right)$ 計算後，最後的結果應為 $(1, 0, 0, 1, 0)$ 。

根據以上得到的結果，對於 $(-8, 3)$ 的行矩陣，其內容為 $(0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0)^T$

5.4 計算平方根

由上一節的結果，讓我們可以找到一個集合 $S = \{(a, b)\}$ ，使得 $\prod_{(a,b) \in S} (a + bm) = y^2$ 且 $\prod_{(a,b) \in S} (a + b\theta) = \delta = \beta^2$ ，由於 $\prod_{(a,b) \in S} (a + bm)$ 的每一個質因數都已知，所以可以很容易求得 y 值。但要如何求出 β ，就比較困難了。

由於我們不知道同態函式 $\varphi: \mathbb{Z}[\theta] \rightarrow \mathbb{Z}/n\mathbb{Z}$ 是什麼，但是可以利用 φ 的性質求出 x 。不過 β 不一定屬於 $\mathbb{Z}[\theta]$ ，但我們可以利用性質 5.5，保證 δ 的平方根必屬於 $\mathbb{Z}[\theta]$ 。

性質 5.5： 令 $f(x) \in \mathbb{Z}[x]$ 是個首一 (monic) 且不可分解 (irreducible) 的多項式， $\theta \in \mathbb{C}$ 為 $f(x)$ 的根，令 I 是在 $\mathbb{Q}(\theta)$ 中的環，且 $\beta \in I$ ，則 $f'(\theta)\beta \in \mathbb{Z}[\theta]$ 。

因此，我們把 δ 的值重新定義為 $(f'(\theta))^2 \times \prod_{(a,b) \in S} (a + b\theta)$ ，則 β 必定屬於 $\mathbb{Z}[\theta]$ 。若 $\beta = a_{d-1}\theta^{d-1} + a_{d-2}\theta^{d-2} + \cdots + a_1\theta + a_0$ ，這時就可利用 $\varphi(\theta) \equiv m \pmod{n}$ 和 $\varphi(a) \equiv a \pmod{n}$ 的性質，得到 $x = a_{d-1}m^{d-1} + a_{d-2}m^{d-2} + \cdots + a_1m + a_0$ ，最後再把 x 模 n 得到我們所需要的值。

由於直接計算 $x \bmod n$ 是十分沒有效率的，我們可以利用中國餘式定理加速計算 $x \bmod n$ 的值。首先，要找到 d 個質數 p ，使得 $f(x)$ 在 $\mathbb{Z}/p\mathbb{Z}$ 中是不可分解的，我們可以利用性質 5.6 來幫忙找出我們所需要的 p 。

性質 5.6： 在 $\mathbb{Z}/p\mathbb{Z}[x]$ 中，多項式 $x^p - x$ 可以被分解為

$$x^p - x = \prod_{i=0}^{p-1} (x - i)$$

因此，在 $\mathbb{Z}/p\mathbb{Z}[x]$ 中，若 $\gcd(x^p - x, f(x)) = 1$ ，則 $f(x)$ 在 $\mathbb{Z}/p\mathbb{Z}[x]$ 中是不可分解的；反之，若 $\gcd(x^p - x, f(x)) \neq 1$ ，則 $f(x)$ 在 $\mathbb{Z}/p\mathbb{Z}[x]$ 中是可分解的。

例 5.6：

令 $f(x) = x^3 + 15x^2 + 29x + 8$ ，檢查 $f(x)$ 在模 $p = 9929$ 時是否是不可分解。

首先計算 $x^p - x = x^{9929} - x \equiv 7449x^2 + 4697x + 5984 \pmod{f(x)}$

計算 $\gcd(7449x^2 + 4697x + 5984, x^3 + 15x^2 + 29x + 8) = 1$

所以在模 9929 下 $f(x)$ 是不可分解的。

接下來計算 $\delta_p = f'(\theta)^2 \times \prod (a + b\theta) \pmod{p}$ ， δ_p 代表著 δ 在模 p 下的值，而且 $\delta_p = \beta_p^2$ ，其中 β_p 代表著 β 在模 p 下的值。以利用性質 5.7，我們可以找出 β_p 的值。

性質 5.7：令 \mathbb{F}_q 是一個有限體，其中 $q = p^d$ ，而 p 是奇質數。令存在正數

整 r 和正奇數 s 使得 $q - 1 = 2^r \cdot s$ 。若 $\eta \in \mathbb{F}_q^*$ 是非二次剩餘，則

$\eta^s \in \mathbb{F}_q^*$ 的 order 為 2^r ，並會存在一個 \mathbb{F}_q^* 的 Sylow 2-subgroup

$S_{2^r} = \{1, \eta^s, \eta^{2s}, \eta^{3s}, \dots, \eta^{(2^r-1)s}\}$ ；在 S_{2^r} 必能找到 $\zeta \in S_{2^r}$ ，使得

$\zeta^2 \equiv \delta_p^s \pmod{p}$ 。

令 $\beta_p = \delta_p^{\frac{s+1}{2}} \zeta^{-1} \pmod{p}$ ，則 $\beta_p^2 \equiv \left(\delta_p^{\frac{s+1}{2}} \zeta^{-1} \right)^2 \equiv \delta_p^{s+1} \zeta^{-2} \equiv \delta_p \pmod{p}$ ，因此

可以求出 β 在模 p 下的值。但要如何在 \mathbb{F}_q 中找到一個非二次剩餘的多項式，我們可以借助性質 5.8 來找出來。

性質 5.8: $f(x)^{\frac{p^d-1}{2}} \pmod{p}$ 等於 1 若且為若 $f(x)$ 在 \mathbb{F}_q 中是一個二次剩餘的多項式； $f(x)^{\frac{p^d-1}{2}} \pmod{p}$ 等於 -1 若且為若 $f(x)$ 在 \mathbb{F}_q 中是一個非二次剩餘的多項式。

所以我們可以檢驗 $(x+j)^{\frac{p^d-1}{2}} \pmod{p}$ 的結果，其中 j 的值由 1 開始遞增進行測試，直到 $(x+j)^{\frac{p^d-1}{2}} \equiv -1 \pmod{p}$ 為止。

令存在質數 p_1, p_2, \dots, p_d ，使得 $f(x)$ 在 $\mathbb{Z}/p_i\mathbb{Z}$ 中是不可分解的，令 $z \equiv \sum_{i=1}^d a_i x_i P_i \pmod{P}$ ，其中 $x_i = \beta_{p_i}(m)$, $P = \prod_{i=1}^d p_i$, $P_i = P/p_i$, $a_i \equiv P_i^{-1} \pmod{p_i}$ ，則 $z \equiv x_i \pmod{p_i}$ ，因此 $x \equiv z \pmod{P}$ 。 x 也就可以表示成 $x = z - rP$ ，其中

$$r = \left\lfloor \frac{1}{2} + \frac{z}{P} \right\rfloor, \text{ 而且 } \frac{z}{P} = \frac{\sum_{i=1}^d a_i x_i P_i}{P} = \frac{\sum_{i=1}^d a_i x_i \frac{P}{p_i}}{P} = \frac{\sum_{i=1}^d a_i x_i}{p_i}, \text{ 所以得到 } r = \left\lfloor \frac{1}{2} + \sum_{i=1}^d \frac{a_i x_i}{p_i} \right\rfloor。$$

而我們要求的最終結果是 $x \pmod{n}$ ，則可利用

$$x \pmod{n} = z \pmod{n} - rP \pmod{n} = \sum_{i=1}^d a_i x_i P_i \pmod{n} - rP \pmod{n}$$

來求得 $x \pmod{n}$ 的值。

總合上述結果，我們可以依照以下步驟找到 x

(1) 令 i 等於 0。

(2) i 的內容加 1。找到質數 p_i ，使得 $f(x)$ 在 $\mathbb{Z}/p_i\mathbb{Z}[x]$ 中是不可分解的。令 $g(x) = x^{p_i} - x \bmod f(x)$ ，若在模 p_i 中， $\gcd(g(x), f(x)) = 1$ ，則 $f(x)$ 在 $\mathbb{Z}/p_i\mathbb{Z}[x]$ 中是不可分解的。否則就繼續檢驗另一個 p 值。

(3) 計算 $\delta_{p_i} = f'(\theta)^2 \times \prod (a + b\theta) \pmod{p_i}$ 。

(4) 要在 $\mathbb{Z}/p_i\mathbb{Z}[x]$ 中找到一個非二次剩餘的多項式。 j 由 1 開始依序代入 $(x+j)^{\frac{p_i^d-1}{d-1}} \bmod p_i$ ，檢查是否為 -1 。若是，則表示 $x+j$ 是非二次剩餘；若不是，則把 j 值加 1 繼續測試。

(5) 求出 ζ_i 的值。首先，計算 $p_i^d - 1 = 2^{r_i} s_i$ ，其中 s_i 是奇數。令

$$\zeta_i \in S_{2^{r_i}} = \{1, \eta_i^s, \eta_i^{2s}, \eta_i^{3s}, \dots, \eta_i^{(2^{r_i}-1)s}\}, \text{ 找出 } \zeta_i \text{ 值, 使得}$$

$$\zeta_i^2 \equiv \delta_{p_i}^s \pmod{p_i}。$$

(6) 計算 $\beta_{p_i} = \delta_{p_i}^{\frac{s+1}{2}} \zeta_i^{-1} \pmod{p_i}$ ，若 i 值小於 d 時，回到第 2 步，否則進到下一個步驟。

(7) 令 $P = \prod_{i=1}^d p_i$ ， $P_i = P / p_i$ ， $a_i \equiv P_i^{-1} \bmod (p_i)$ ， $x_i \equiv \beta_{p_i}(m) \pmod{p_i}$ ，

$$\text{計算 } r = \left\lfloor \frac{1}{2} + \sum_{i=1}^d \frac{a_i x_i}{p_i} \right\rfloor。 \text{ 則 } x \equiv \sum_{i=1}^d a_i x_i P_i - rP \pmod{n}。$$

由於我們在求 x 的值時，是針對 $f'(\theta)^2 \times \prod_{(a,b) \in S} (a + b\theta)$ 分解，所以要求 y 值

是，必須針對 $f'(m)^2 \times \prod_{(a,b) \in S} (a + bm)$ 分解，由於我們已知道 $f'(m)^2 \times \prod_{(a,b) \in S} (a + bm)$

的因數，所以可以很快地分解出 y 值。

求出 x, y 值後，先檢驗在模 n 下， x 是否等於 $\pm y$ ，若 $x \equiv \pm y \pmod{n}$ ，則表示無法分解，就必須重新分解；否則， $n = \gcd(x + y, n) \times \gcd(x - y, n)$ 。

第6章 實驗結果與分析

本篇論文的實驗部份是在一台 Linux Server 上進行的，利用通用型數域篩選演算法（General Number Field Sieve）對於 61 位的整數作因數分解，並藉由調整「 $a + bm$ 和 $a+b\theta$ 中的 a 之篩選範圍 M 」以及「有理數分解基底（Rational Factor Base, 簡寫為 RFB）的個數」這兩個實驗參數，以觀察其對分解所需時間的影響。最後，針對實驗所得之數據進行分析，並建議該如何調整其參數以增加分解的效率。

6.1 實作環境

6.1.1 硬體環境

- 中央處理器：Pentium 4 2.54G （1 顆）
- 記憶體：DDR RAM 1G （512 MB 2 條）
- 主機板：FSB 533

6.1.2 軟體環境

- 作業系統：Redhat 9.0
- 編譯器：GCC 3.2.3
- 大數運算使用的函式庫：gmp 4.1.2

6.2 系統架構

以下用程式流程圖來描繪系統架構。

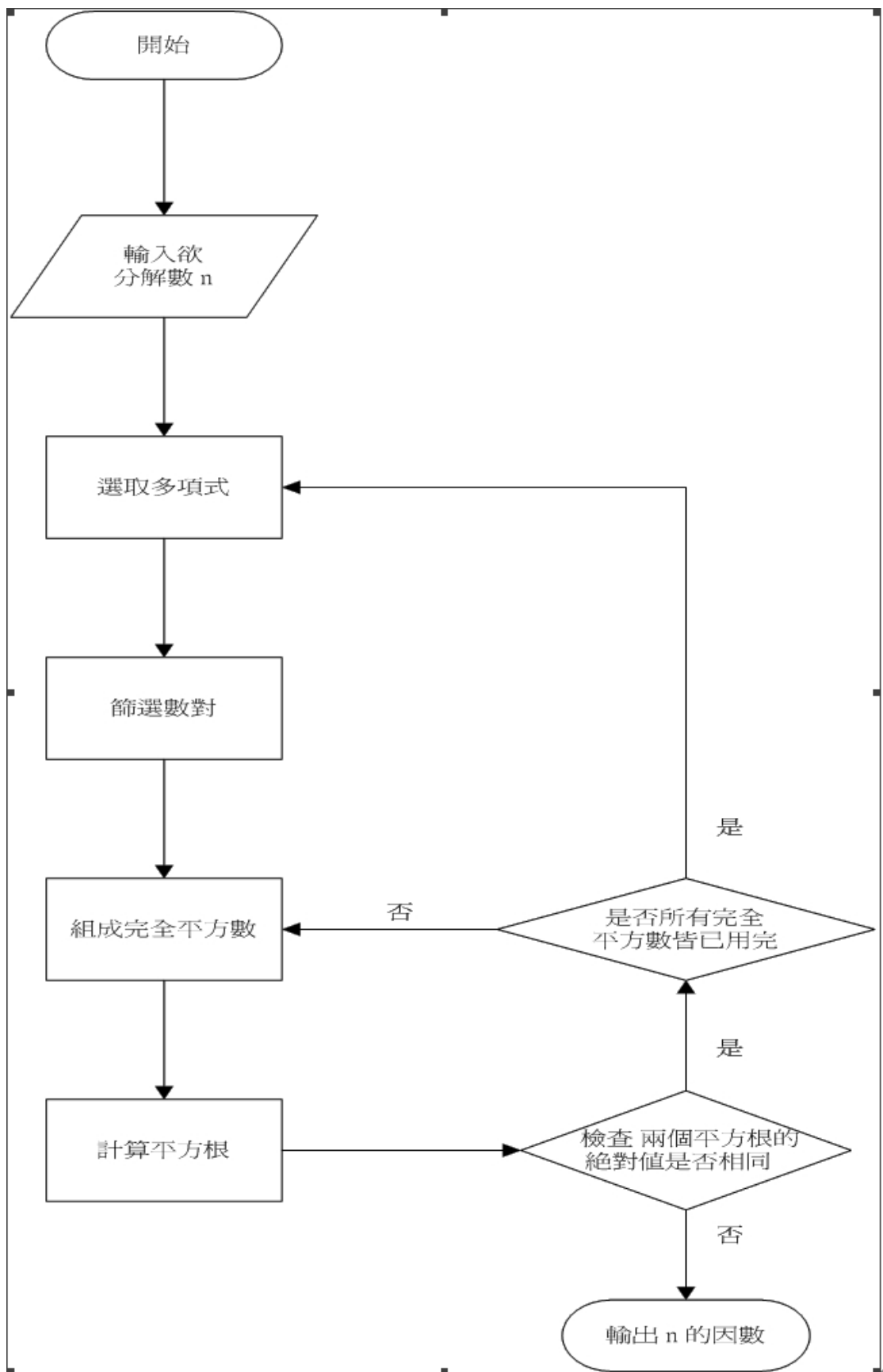


圖 6-1 GNFS 的程式流程圖

6.3 實驗參數

以下列出在本實驗中所使用的參數及其代號

- n ：欲分解的數。
- d ：多項式 $f(x)$ 的最高冪次。
- m ：正整數 m ，使得多項式 $f(m) \equiv 0 \pmod{n}$ 。
- c_i ：多項式 $f(x)$ 的係數。
- #RFB：有理數分解基底(Rational Factor Base)的個數。
- #AFB：代數數分解基底(Algebraic Factor Base)的個數。
- #QCB：二次特徵基底(Quadratic Character Base)的個數。
- M ： a 篩選的範圍($-M \sim M$)。

6.4 實驗結果

在本實驗中，所用固定的實驗參數如下：

- $n = 124144515376516209037603246156$
 $4730757085137334450817128010073$
- $d = 3$
- $m = 81591306743478409$
- $c_3 = 2285577075$
 $c_2 = 1236944707418$
 $c_1 = -137518782158383$
 $c_0 = 2238328747672587$
- #AFB = 10000

當實驗參數設定為上列所列的值時，調整「 $a + bm$ 和 $a + b\theta$ 中的 a 之篩選範圍 M 」以及「RFB 的個數」這兩個參數，得到表 6.1 的數據。

$M \backslash \text{RFB}$	8000	10000	12000
500000	13776	7743	5739
600000	9217	5820	5254
700000	7162	5479	4623
800000	7530	4644	5185
900000	6038	5138	3980
1000000	5086	5571	4355
1100000	5549	4217	4716
1200000	6391	4542	5082
1300000	4360	4879	5442
1400000	4671	5231	5817
1500000	4955	5531	6162
1600000	5254	5848	6513

表 6-1 調整篩選範圍和 RFB 的數據結果

表 6-1 中，每一行代表當 RFB 固定時，改變 a 的篩選範圍對於篩選出足夠數對所需時間的影響；每一列代表當 a 的篩選範圍固定時，改變 RFB 對於篩選出足夠數對所需時間的影響；下面是表 6-1 的數據比較圖。

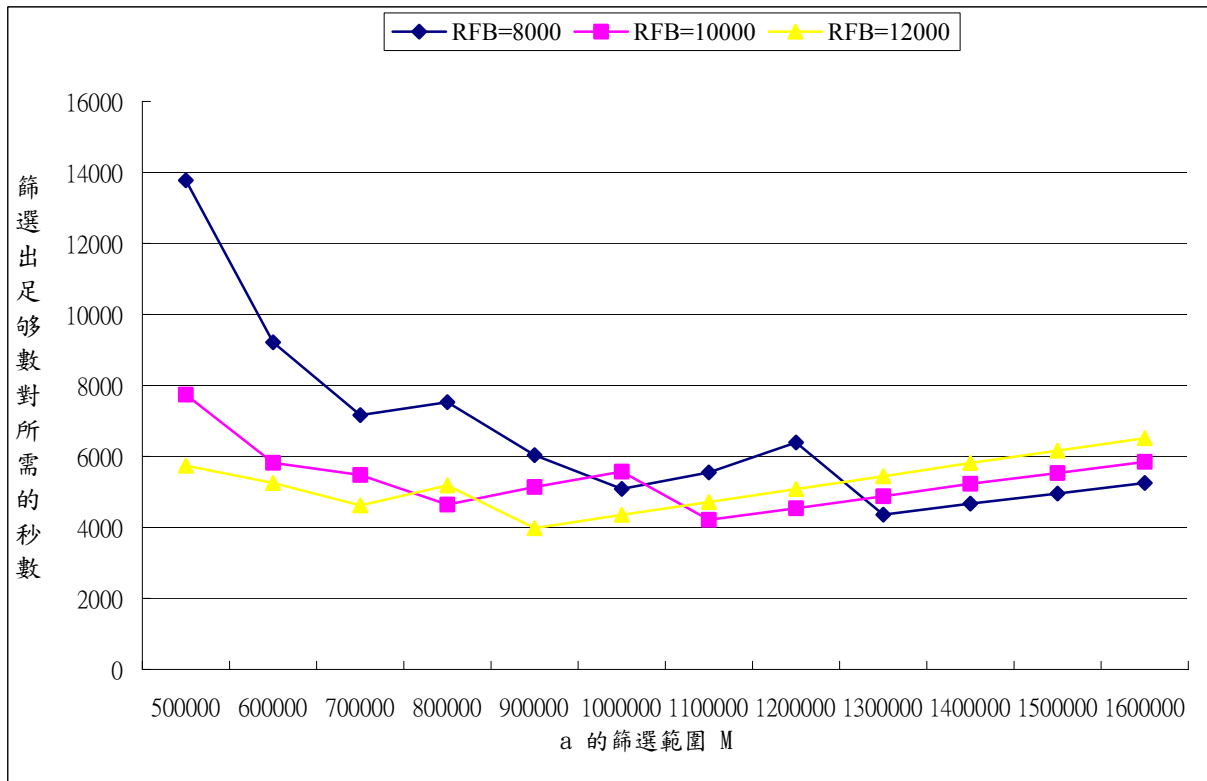


圖 6-2 調整 a 的篩選範圍和 RFB 個數的數據比較圖

6.5 數據分析

對固定 b 值和 RFB 個數的篩選而言， a 的篩選範圍越大，越能找到符合篩選條件的數對；但對整體的篩選時間而言，若篩選範圍超過某一個值後，雖然符合篩選條件的數對會比較小的篩選範圍多一點，但是，不符合篩選條件的數對會比較小的篩選範圍多更多，因此導致整體篩選的效率降低。

對固定 a 和 b 值的篩選而言，RFB 的個數越多，越能找到符合篩選條件的數對；但相對地，所需找到 smooth 的數對也必須增加。因此，對整體的篩選時間而言，當 RFB 的個數變多時，在較小的篩選範圍所需的篩選時間會比較少，但在超過某個的篩選範圍時，全部所需的篩選時間反而會比較多。

在調整參數的過程中發現，對固定 b 值和 RFB 個數的篩選而言，若 a 的篩選範圍變大，但篩選效率卻降低時，這時候就要考慮換調整 RFB；對固定 a 和 b 值的篩選而言，若 RFB 的個數越多，但篩選效率卻變差時，這時候就要考慮換調整 a 的篩選範圍。藉由如此反覆調整，以找出最佳的分解效率。

第7章 結論

在通用型數域篩選演算法分解法中，花費時間最多的部份是在於篩選出足夠多的滿足在有理數分解基底和代數數分解基底均為 smooth 條件的數對，而影響篩選速度的參數有多項式 $f(x)$ 的係數選取、有理數分解基底的個數數量、代數數分解基底的個數數量、二次特徵基底的個數數量以及 a 篩選的範圍大小，而這些參數的選擇，至今還沒有一個很好的決定方法，所以在這方面還有許多值得研究的地方。

雖然篩選這部份會花費很多的時間，但對於不同的 b 值的篩選，其結果是獨立的，但這部分卻是這個分解法中最容易平行處理的。因此，許多大型的因數分解計畫，都是由主機將不同的 b 值分配給不同的電腦同時進行篩選的動作，再把結果回傳給主機，以縮短篩選所需的時間。而如何利用平行運算以加速分解的效率，這是本篇論文值得改進的方向之一。

就分解 100 位數以上的 RSA 數而言，GNFS 是最快的分解方法。目前 RSA 分解的保持紀錄為 RSA-576 (174 decimal digits)[19,20]，這是在 2003 年 12 月 3 日，由德國的 Federal Agency for Information Technology Security (BIS) 成功地分解出來，其所用的分解方法正是 GNFS。所以，分解更高位數的整數，也是本篇論文值得改進的方向之一。

參考文獻

- [1] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the theory of NP-Completeness*, W.H. freeman and Co., 1979.
- [2] R.P. Brent, *Primality Testing and Integer Factorization*, Proceeding of Australian Academy of Science Annual General Meeting Symposium on the Role of Mathematics in Science, Canberra, 1991, 14-26
- [3] Song Y. Yan, *Number Theory for Computing*, 2nd ed., Springer-Verlag, Berlin, 2002
- [4] Hans Riesel, *Prime numbers and computer methods for factorization*, Birkhäuser, Boston 1985, 2nd ed. 1994
- [5] David M. Bressoud, *Factorization and primality testing*, Undergraduate Texts in Mathematics, Springer-Verlag, New York, 1989.
- [6] J. M. Pollard, *A Monte Carlo method for factorization*, BIT. V15(3), 1975 , pp. 331-334
- [7] J. M. Pollard, *Theorems on factorization and primality testing*, Proc. Camb. Phil. Soc., v76(2) , 1974 , pp 521-528
- [8] H. W. Lenstra, Jr., *Factoring integers with elliptic curves*, Annals of Math., v126(3), 1987, pp.649-673
- [9] J. Dixon, *Asymptotically fast factorization of integers*, Mathematics of computation, vol. 36, no. 153, pp. 255-260, 1981
- [10] M. A. Morrison, J. Brillhard, *A method of factoring and factorization of F_7* , Math of Comp., v29, 1975, pp. 183-205
- [11] C. Pomerace, *The quadratic sieve factoring algorithm*, Advances in Cryptology: Proceedings of EUROCRYPT 84(1984), pp. 169-182.
- [12] R.D. Silverman, *The multiple polynomial Quadratic Sieve*, Mathematics of Computation 48 (1987), pp329- 339
- [13] Brandt Kurowski, *The Multiple Polynomial Quadratic Sieve*, <http://brandt.kurowski.net/projects/mpqs/paper/mpqs.ps>

- [14] P. L. Montgomery, *A survey of modern integer factorization algorithms*, CWI Quarterly 7 (1994), pp337–366.
- [15] Matthew E. Briggs, *An introduction to the general number field sieve*, M.S. Thesis, Virginia Polytechnic Institute, Blacksburg, Virginia, 1998.
- [16] Michael Case, *A Beginner's Guide To The General Number Field Sieve*, <http://islab.oregonstate.edu/koc/ece575/03Project/Case/paper.pdf>
- [17] David Stephenson, *Factoring Very Large Numbers*, <http://citeseer.nj.nec.com/stephenson96factoring.html>
- [18] Joel Dreibelbis, *Implementing the General Number Field Sieve*, <http://www.cs.rit.edu/~jdd5747/msPaper.ps.gz>
- [19] Franke, J. , *RSA576*, Privately circulated email reposted to primenumbers Yahoo! Group, <http://groups.yahoo.com/group/primenumbers/message/14113>
- [20] Eric W. Weisstein, *RSA Number*, From MathWorld--A Wolfram Web Resource. <http://mathworld.wolfram.com/RSANumber.html>
- [21] 賴溪松、韓亮、張真誠，*近代密碼學及其應用*，松崗書局，2000
- [22] 趙文敏，*數論淺談*，協進，1985