Advancements in TMTOs

Steve Thomas “Sc00bz”

Passwords^12

TobTu.com/passwords12

Time-Memory Trade-Offs

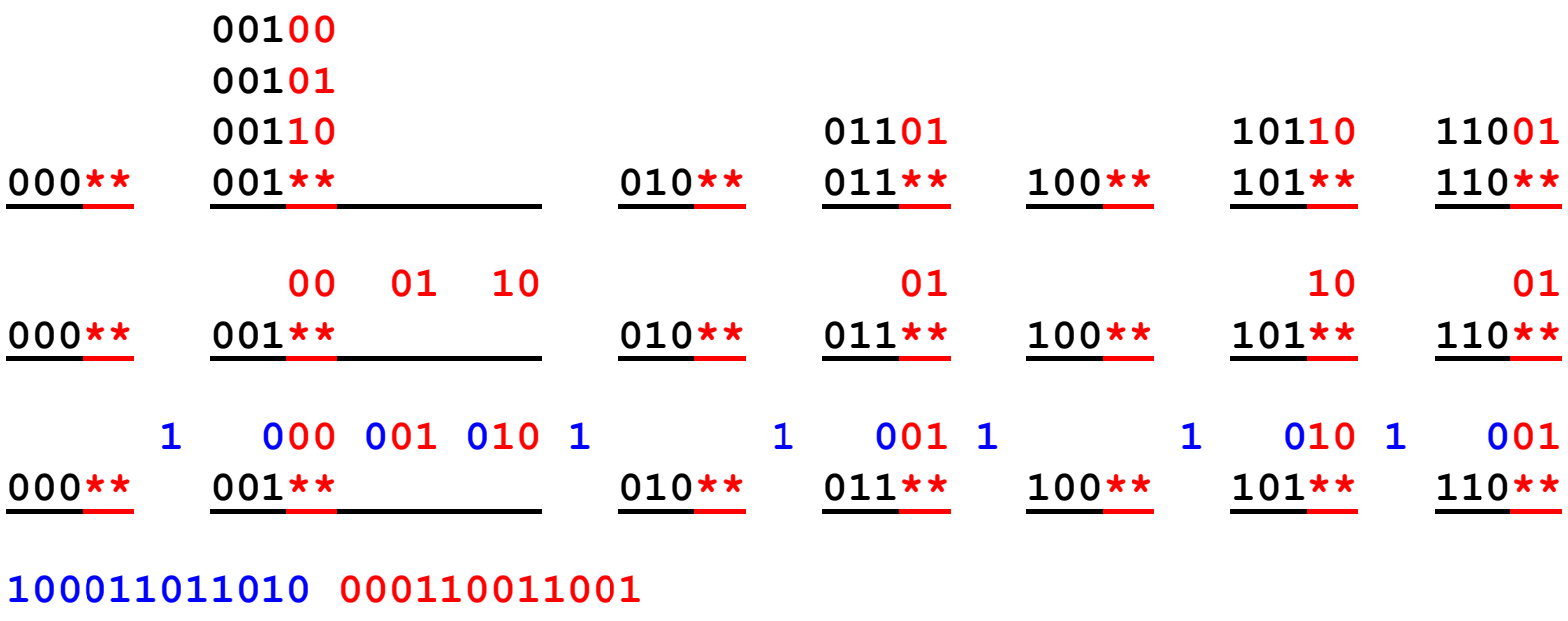
- Brute force
- Hash tables
- Chained tables

Minimize Memory

- Minimal Perfect Hash Function (MPHF)
  - Compressed hash-and-displace (CHD)
- Elias-Fano Encoding
  - Monotone increasing sequence
  - SuffixBits = max(floor($\log_2$(Max / Count)), 0)
  - Buckets = ceil(Max / 2 ^ SuffixBits) ≈ Count

Elias-Fano Encoding

{4, 5, 6, 13, 22, 25} in decimal
{00100, 00101, 00110, 01101, 10110, 11001} in binary



Minimal Perfect Hash Function

- Compressed hash-and-displace (CDH)
  - λ – keys/bucket
  - Process largest bucket first
  - Fredriksson-Nikitin encoding
    - Elias-Fano encoded bit offsets into the data stream
    - Data stream
      - 0 – “”, 1 – “0”, 2 – “1” , 3 – “00”, 4 – “01”, 5 – “10”…

Lossy Hash Table (LHT)

- MPHF
  - Mini index
  - Index the full hash
  - Store a password range
  - Worst case is 2x the average case

Lossy Hash Table (LHT)

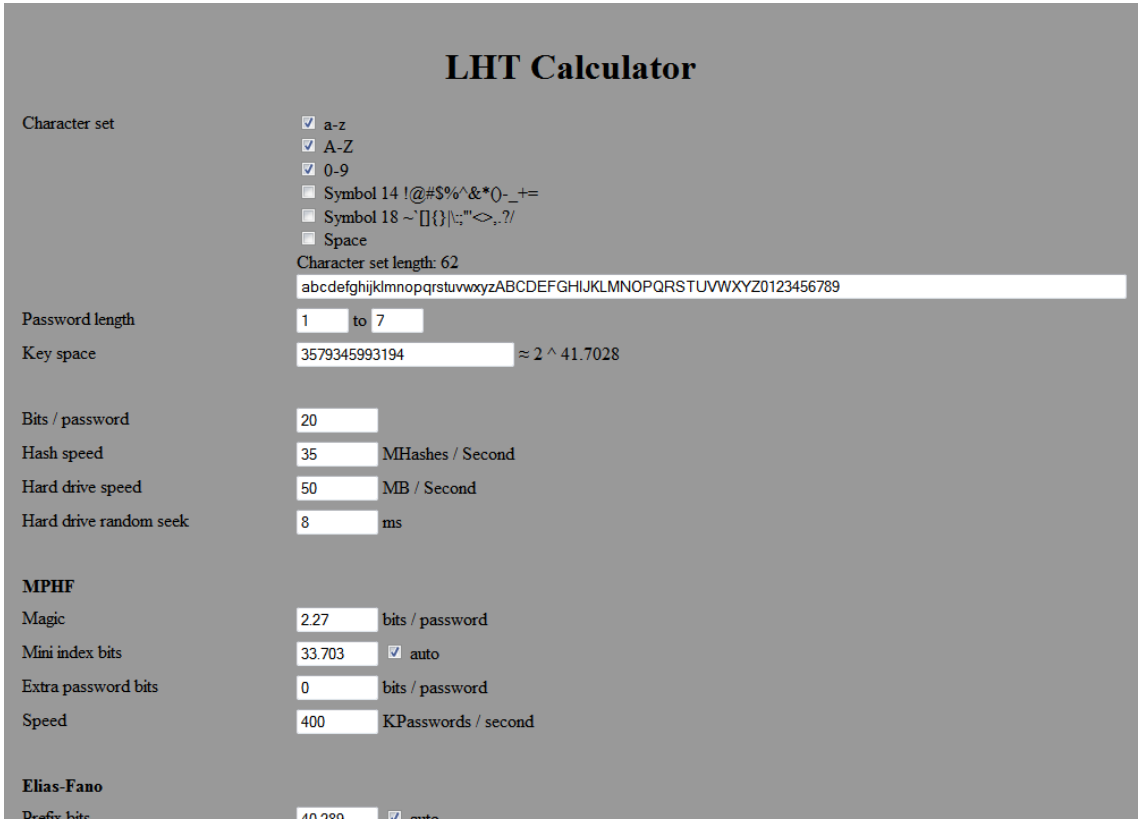
- Elias-Fano Encoding
  - Mini index
  - Index part of the hash
    - HashBits < log2(KeySpace)
      - SuffixBits = 0
      - Buckets < Count
  - Store a password range
  - Worst case is 4x-8x the average case

LHTs are “Instant”

- Best for web services
  - MD5
  - NTLM
  - PDF/Excel/Word
    - 3.5 TB takes 55 ms
    - Patented?
      - Yes but invalid
        - Try at your own risk8

LHT Calculator

TobTu.com/lhtcalc.php



Chained Tables

- Hellman Tables (Classical Tables)
- Distinguished Points (DP)
- Rainbow Tables (RT)
- Varying/Variable Distinguished Points (VDP)
- Combinations of [V]DP and Rainbow, Hellman, and chained
- Spoiler any DP is worse than RT because of inverse relationship with success rate and DP work factor

Reduction Functions

- Divide (RCrack)
  - 32 bit floating point multiply (4x speed GPUs)
- Look up tables (GRT)
- Fixed point multiply (FPM)
- Dictionary
- Markov

Start Points: Random vs Sequential

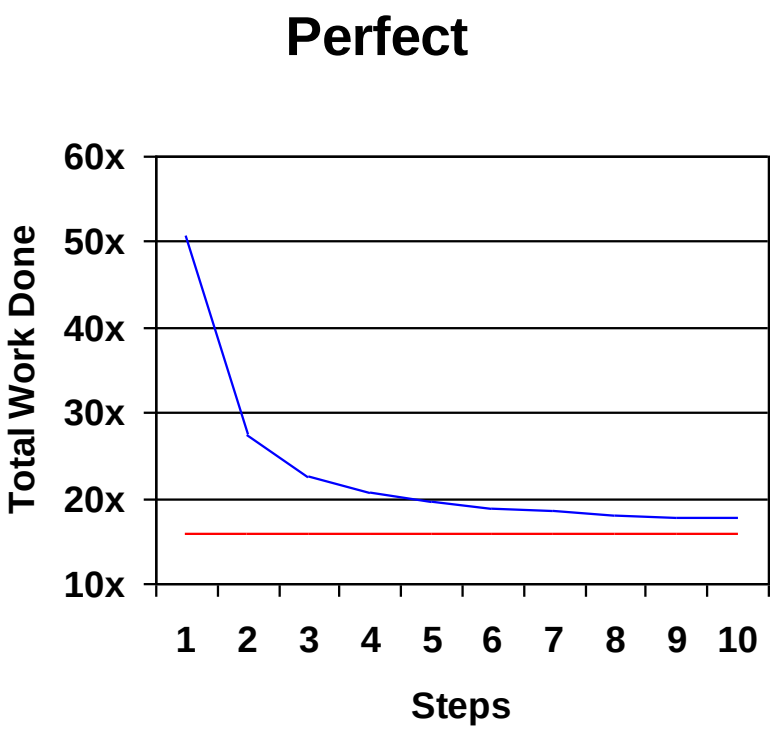
- Size
- Duplicates
- Key space coverage

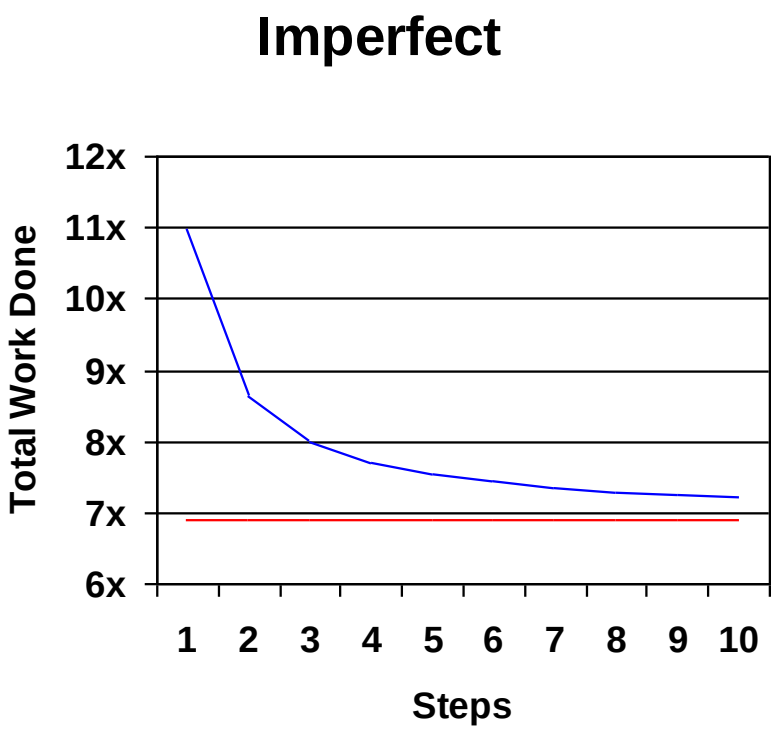
Start Points: Random vs Sequential

- Stop using random start points

Perfect vs Imperfect

Step Generation





RT File Formats

- DIRT – MPHF
- IRT – Elias-Fano
- RTI2 – 8 bit prefix index
- RTC – Linear regression
- RTI – 8 bytes / chain + 11 byte prefix index
- GRT2 – Packed chain + 12 byte prefix index
- RT – 16 bytes / chain
- GRT – 32 bytes / chain + 12 byte prefix index

Perfect vs Imperfect

- Compared
  - Key spaces 95#1-7 and 95#1-8
  - 10k, 20k, 50k, 100k chain lengths
  - DIRT and IRT file formats
- Generation
  - 4.61x more work to generate perfect
- Step Generation
  - 2 Steps: 3.16x more work to generate perfect
  - 3 Steps: 2.82x more work to generate perfect
  - 4 Steps: 2.68x more work to generate perfect
  - Limit: 2.31x more work to generate perfect

Perfect vs Imperfect

- Size (varies with key space)
  - Imperfect DIRT is 50% larger than perfect DIRT
  - Imperfect IRT is 33% larger than perfect IRT
    - Decreases with larger chain lengths (25%)
  - Perfect IRT is 20% larger than perfect DIRT
    - Increases with larger chain lengths (30%)
  - Imperfect IRT is 8% larger than imperfect DIRT
    - Increases with larger chain lengths (12%)
- Time
  - Imperfect DIRT takes 20% more time than perfect DIRT
  - Imperfect IRT takes 23% more time than perfect IRT
  - Perfect DIRT takes 5% more time than perfect IRT
  - Imperfect DIRT takes 3% more time than imperfect IRT

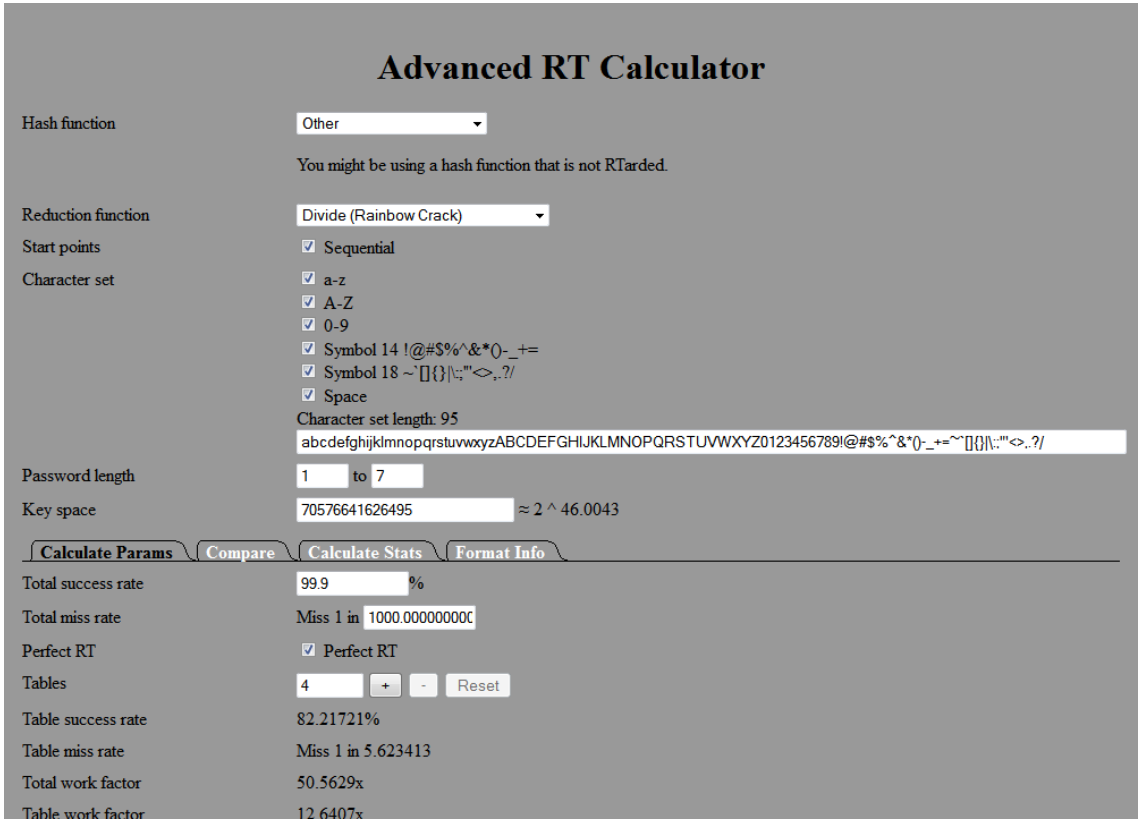
Checkpoints

100% Rainbow Table

- Patented?
  - Yes and valid
- Work around
  - Yes and better*
- Full sort instead of ***
- Store passwords in a LHT (MPHF) instead of a list of passwords in 256 files.

Advanced RT Calculator

TobTu.com/rtcalc.php



Perfect Hellman Tables

- Can’t use DIRT format (MPHF)
- Generation
  - No more bloom filters
  - 1/Nth reduction of memory

Thank You

- Questions?