

## §7 Code Tweaking

- optimization
  - utilize available resources more efficiently
  - do not change the intended behaviour
- resources
  - running time
  - internal and external memory space
  - interaction in the execution of the system
- time/space/communication

## Observations on optimization

1. consider the most dominating factor affecting the run time
2. compiler is programmers' friend (and not their fiend)
3. optimization  $\rightarrow$  obscured implementation  $\rightarrow$  reduced maintainability
4. 20/80 rule: find and refine the hot spots
5. inherent complexity: at some point optimization becomes pessimization

## Bit fiddling

- bit  $b \in \mathbf{B} = \{0, 1\}$
- $n$ -bit word  $w \in \mathbf{B}^n$ 
  - nibble:  $n = 4$
  - byte (or octet):  $n = 8$
- indexing
  - $w = b_{n-1} b_{n-2} \dots b_1 b_0$
  - least significant bit (LSB):  $b_0$
  - most significant bit (MSB):  $b_{n-1}$

## Grouping bits

- block of consecutive bits
  - position  $s$
  - length  $l$
- selection of bits
  - mask characterizes the selection with 1-bits
- array of buckets

## Bit-parallel routines

- algorithmic thinking is useful also at the bitwise level
  - divide-and-conquer
  - dynamic programming
  - sieve iteration
- used in restructuring the computation
  - based on insights into the mathematical identities
  - hard to give general rules

## Sets, power sets, and Gray codes

- assumptions
  - enumerable universe of discourse
  - characteristic function
- power set: enumerate all possible subsets
- Gray code
  - minimize the change in the bit encoding when adding/removing elements to/from the set
  - binary-reflected  $n$ -bit Gray code

### Example: $G(4)$ and $C(4)$

0	0000	—	8	1100	3
1	0001	0	9	1101	0
2	0011	1	10	1111	1
3	0010	0	11	1110	0
4	0110	2	12	1010	2
5	0111	0	13	1011	0
6	0101	1	14	1001	1
7	0100	0	15	1000	0

### Bit reversal

- conversion approaches
  - naïve: looping the bits one by one
  - bit parallelism: operating with words
  - preprocessed data: reducing run-time operations
- uses for bit reversal
  - fast Fourier transform
  - quasi-random numbers

Example: dissolve