CP8201/CPS815 Advanced Algorithms

Assignment 3

Rohaan Ahmed
PhD Student
Department of Computer Science
rohaan.ahmed@ryerson.ca

November 10, 2020

Instructor: Dr. Javad (Jake) Doliskani
November 10, 2020

1 Algorithm to find a $1/2^t$ approximation of the inverse of f

Let g be the inverse of f and h be the $1/2^t$ approximation of g. Then, the following equations are true based on the information provided in the question:

Let $h(g)$ be a $1/2^t$ approximation of g. Then:

$$h(g) = (\frac{1}{g} - f) \bmod x^{2^t} = 0 \bmod x^{2^t} \tag{0.1}$$

Plugging into Newton-Raphson:

$$g_{n+1} = g_n - \frac{h}{h'} \tag{0.2}$$

$$g_{n+1} = \frac{\frac{1}{g_n} - f}{\frac{-1}{g_n^2}} \tag{0.3}$$

$$g_{n+1} = 2g_n - fg_n^2 \tag{0.4}$$

Therefore:
For $n = 0$:

$$fg_0 = f(0)g_o = (1)(1^{-1}) = 1 \tag{0.5}$$

For $n = i + 1$, i.e., $g_1 \bmod x^2, g_2 \bmod x^{2^2}, ...$:

$$\begin{aligned} 1 - fg_{i+1} &= 1 - f(2g_i - fg_i^2) \bmod x^{2^{i+1}} && (0.6)\\ &= 1 - 2fg_i - f^2g_i^2 \bmod x^{2^{i+1}} && (0.7)\\ &= (1 - fg_i)^2 \bmod x^{2^{i+1}} && (0.8)\\ & && (0.9) \end{aligned}$$

Thus, to get a $1/2^t$ approximation of g, we iterate until $n = t$:

$$g_t = (2g_{t-1} - fg_{t-1}^2) \bmod x^{2^t} \tag{0.10}$$

Function: ***inverse_approximation(***$f, f(0) = 1, t$***)***

$g_0 = 1$ {`// from Equation 0.5`}
for $i = 1, ..., t$ **do**
  $g_i = (2g_{i-1} - fg_{i-1}^2) \bmod x^{2^i}$ {`// from Equation 0.10`}
end for
return $h = g_t$

$h = g_t$ will be a $1/2^t$ approximation of g, the inverse of f.

2 Complexity of Algorithm

At each timestep, the algorithm above performs a polynomial multiplcation between f and g_{i-1}^2.

Operation 1: $(g_{i-1})(g_{i-1})$ gives us $n_1 = (2^{i-1})(2^{i-1}) = 2^{2i-2}$, where $i = 1, ..., t$
$\implies M(2^t + 2^t) = O(M(2^t))$

Operation 2: $(f)(g_{i-1})$ gives us $n_2 = (N)(2^{i-1})$, where $i = 1, ..., t$
$\implies M(t + 2^t) = O(M(2^t))$

Thus, the complexity of the algorithm is $O(M(2^t))$