

GPU optimization of base form 820 - collective artificial spouse retirement plan

Nikolaj Aaes, Niklas Schalck Johansson and Hildur Uffe Flemberg

22-05-2013

1 Abstract

Abstract

2 Introduction

- Why is the project relevant, what is the problem at hand

Scope? [2]

Hvad regner vi med at lseren har af kompetencer

3 Background

SCOPE: Vi krer kun med een kunde ikke flere

- Related work, projects that precedes this project, why do we use the technology described, what is this new technology

3.1 Scope

4 The Math

The algorithm in this project is used to determine the lump sum of money the insurance company needs to possess to be able to pay the insurance holder's spouse in the case of his or her death. The payment to the spouse will be in

the form of a life interest which are disbursed from the time of death of the insurance holder unless the death occurs before the pension age. In that case the money will be disbursed after a grace period determined by the insurance company at the time the insurance was taken out.

We assume that the spouse of the insurance holder is of the opposite gender. If the insurance holder does not have a spouse at the time of his or her death the insurance is forfeited.

The algorithm used here is a 4th order Runge-Kutta solution with a fixed stepsize (indst reference) where we use a series of constants determined before any calculation begins:

Constants

Name	Meaning
τ	The time of death of the insurance holder
r	The pension age
g	The grace period
x	The age of the insurance holder at calculation time ($t = 0$)
t	The time of calculation
h	The Stepsize of the Runge-Kutta solution

τ is expressed as $x + t$. Apart from this there is also a constant k which are determined by τ, r and g in the following manner:

How k is defined

If this statement holds	then k equals
$\tau < r$	g
$r \leq \tau < r + g$	$r + g - \tau$
$r + g \leq \tau$	0

The algorithm can be described as a combination of three models; an outer model that describes the life/death state of the insurance holder, a middle model that describes the married/unmarried state of the insurance holder and an inner model that describes the life/death state of the potential spouse.

NOTE: The description in the three model sections is loosely based on a description for Edlund[1] and some passages are directly translated from this document.

4.1 Outer model

The outer model is expressed as the following equation:

$$\frac{d}{dt}f(t) = r(t)f(t) - \mu_t(x+t)(S_{x+t}^d(t) - f(t))$$

Where S_{τ}^d is the DDSFALDSSUM that for a τ year old at the time t is needed to cover the payment from the insurance company, $\mu_t(x+t)$ is the mortality rate for a $(x+t)$ year old and $r(t)$ is the interest rate function. In this project the interest rate function returns the constant 0.05.

The differential equation is solved from $t = 120 - x$ to $t = 0$ with the boundary condition $f(120 - x) = 0$

4.2 Middle model

The middle model is used to calculate $S_{x+t}^d(t)$ and is expressed with the following equation:

$$S_{\tau}^d(t) = \begin{cases} g_{\tau} \int f(\eta|\tau) a_{[\eta]+g}^I(t) d\eta & \tau \leq r \\ g_{\tau} \int f(\eta|\tau) a_{[\eta]+r+g-\tau}^I(t) d\eta & r \leq \tau \leq r+g \\ g_{\tau} \int f(\eta|\tau) a_{[\eta]}^I(t) d\eta & r+g \leq \tau \end{cases}$$

Where g_{τ} is the propability that a τ year is married and $f(\eta|\tau)$ is the probability distribution for, that a τ year old is married to a η year provided that the τ year old is married. This equation can be rewritten to this form:

$$\frac{d}{dn}f(\eta) = -g_{\tau}f(\eta|\tau)a_{[\eta]+k}^I(t)$$

here the DET-DER-TAL-SOM-LIGNER-EN-EKSPONENT-MEN-ER-NEDE-I-STEDET-FOR-OPPE for a is rewritten to $[\eta]+k$ where k can fall into three different categories as shown in THE-TABLE-CONSTANTS.

This differential equation is solved from $\eta = 120$ to $\eta = 1$ with the boundary condition $f(120) = 0$

4.3 Inner model

The inner model is used to calculate $a_{[\eta]+k}^I(t)$ and is expressed with the following equation:

$$\frac{d}{ds}f(s) = r(t+s)f(s) - 1_{s \geq k} - \mu_{t+s}(\eta+s)(0-f(s))$$

Where t , η and k are constants and $\mu_{t+s}(\eta+s)$ is the mortality rate for a $(\eta+s)$ year old.

This differential equation is solved from $s = 120 - \eta$ to $s = 0$ with the boundary condition $f(120 - \eta) = 0$

4.4 Runge-Kutta 4th order

The Runge-Kutta method is a method for approximating differential equations that builds on Eulers method (INDST REFERENCE) and the midpoint method (INDST REFERENCE). When given a start point and a differential equation one can choose a stepsize and approximate the graph. When given a point (x_n, y_n) , a differential equation f and a stepsize $h > 0$ one can use the Runge-Kutta method to approximate the next point in the following way:

$$\begin{aligned} k1 &= hf(x_n, y_n) \\ k2 &= hf(x_n + \frac{h}{2}, y_n + \frac{k1}{h}) \\ k3 &= hf(x_n + \frac{h}{2}, y_n + \frac{k2}{h}) \\ k4 &= hf(x_n + h, y_n + k3) \end{aligned}$$

$$\begin{aligned} x_{n+1} &= x_n + h \\ y_{n+1} &= y_n + \frac{k1}{6} + \frac{k2}{3} + \frac{k3}{3} + \frac{k4}{6} + O(h^5) \end{aligned}$$

$O(h^5)??$ HVORFOR KAN VI IGNORERE

BESKRIV HVORDAN EN NORMAL KRSEL SERUD = EKSEMPEL
kig evt p en simpson integration nr vi kigger p middle da det egentlig ikke er en differential ligning men et integrale

5 CUDA

en trd skal lave et vist stykke arbejde fr det kan betale sig at oprette den. Der er overhead for at lave context for hver trd s hvis den arbejde trden laver er mindre end at stte overheaden op, giver det ingen mening.

- A more elaborate description of GPGPUs, architecture and CUDA C.

5.1 Parallellisation

De forskellige parallelliserings muligheder: - beskriv teorien - beskriv hvad vi får ud af det i kretid? - beskriv hvorfor vi har/ikke har valgt denne løsning - Kan den kombineres med de andre for bedre performance?

Outer Par

Da man på forhånd kan beregne alle x værdier outer skal bruge i sin kreds og alle $\text{Middle}(x)$ kald i outerDiff kun skal bruge en x værdi er det muligt at regne alle disse middle kald i parallel og gemme hvert middle kalds resultat i et array. Når man derefter skal gå fra startpunkt til næste punkt kan man betragte $\text{Middle}(p.x)$ i $\text{return}(px) * py - GmFemale(x + px) * (\text{Middle}(px).y - py)$; som en udregning der tager konstant tid.

Middle Par

Da man på forhånd kan beregne alle x værdier middle skal bruge i sin kreds og vi ved at hvert step i Middle har udregninger til $k_1, k_2/k_3$ og k_4 i hvert skridt der kun afhænger af x -værdien kan vi regne alle de st af $k_1, k_2/k_3$ og k_4 som middle skal bruge i parallel. Derefter kan de lægges sammen som et gennemsnit i et array og bruger når man vil gå fra et punkt til et nyt. Kaldet $\text{doubley} = p.y + k_1/6 + k_2/3 + k_3/3 + k_4/6$; kan altså betragtes som konstant. Her udregnes hvert st af $k_1, k_2/k_3$ og k_4 dog i en enkelt tråd. Det vil sige at der er en tråd for hvert step.

Inner Par

Man har også muligheden for at parallelisere middles kald til inner der bruges i udregningen af $k_1, k_2/k_3$ og k_4 . Ved at splitte 3 tråde til at udregne de tre værdier kan dette gå tre gange så hurtigt.

Kombination af Outer Par og Middle par

Ved en valgt block size b = antal outersteps og en thread per block tpb kan man splitte hver block til at tage sig af en x -værdi i outer, altså tage sig af et enkelt middle kald. Hvis man forestiller sig at middle skal tage ms steps og $ms \geq tpb$, vil man først udregne $k_1, k_2/k_3$ og k_4 st for de første tpb steps og derefter gå videre til de til næste tpb steps indtil alle steps er udregnet. Det betyder også at der kan være inaktive tråde i den sidste omgang tpb steps og det kan give divergence. Det er muligt at bruge reduction (se cuda bogen) når alle k er blevet regnet ud og man skal lægge $p.y$ til.

Kombination af Middle par og Inner par

Ligesom i den ovenstende løsning lader man hver block handle en x-vrdi for middle og giv hver block tre trde hver af disse trde udregner hver sin k vrdi og hver tredje trd lgger dem sammen.

Kombination af alle tre

Som i kombinationen af outer par og middle par vlger man et antal blokke svarende til antallet af outersteps og en thread per block tpb. Man lader stadig hver blok tage sig af et enkelt kald til Middle med en bestemt x-vrdi. Forskellen kommer nr man i stedet for at lade hvert trd udregne et st af k_1 , k_2/k_3 og k_4 vrdier stter man i stedet 3 trde til at udregne dem i parallel. Dette vil kun have indvirkning hvis block strrelsen er strre end det antal steps middle skal udfre. Hvis ikke bruger man tre gange s mange trde til at s udregnes frst tpb k-vrdier, tre gange hurtigere end normal, dernst de nste tpb k-vrdier, stadig tre gange s hurtigt men i stedet for at stoppe ved middlesteps som normal vil man stoppe ved $\text{middlesteps} * 3$ da der skal bruges 3 gange s mange k-vrdier. Optimalt set skal der alts vre $\text{middlesteps} * 3$ trde i hvert block. Ved en vilkrlig stepsize vil det vre $120 * \text{stepsize} * 3$ trde. Med en stepsize p 1 eller to giver det henholdsvis 360 og 720 trde hvilket stadig er inden for det 1024-trde limit hver block har. Men allerede ved stepsize 3 giver det 1080 trde per block hvilket ikke er legalt.

Hvad har vi valgt

Vi har valgt Kombination af Outer Par og Middle Par for simpelheden. Kombination af alle tre er ikke smart fordi det kun giver optimering nr threads per block er $\text{middlesteps} * 3$ eller over hvilket ikke er legalt med en stepsize over 2. SKRIV NOGET OM HVORFOR VI HAR VALGT KOMBI AF OUTER OG MIDDLE I STEDET FOR MIDDLE OG INNER. HVORFOR ER DET MERE EFFEKTIVIT?

6 Implementation

- A description of the new implementation and design choices I den kode der er nu er insurance holder kvinde og spouse mand ALTID skriv noget om memory bound og compute bound i rapporten

7 Testing

test p forskellige maskiner hvis vi deler steps op i blocks af 32 trde kan der i worst case scenario vre en sidste block med 1 aktiv og 31 inaktive trde. Lav en graf over rest trdene

- A test to ensure that the new implementation produces the same output as the current implementation

8 Benchmarks and Comparison

- Benchmarking and comparison on speed between the current and new implementation

9 Conclusion

Conclusion

10 Reflection / Discussion / Future improvements

improvements: det kan optimeres til at kunne kre flere kunder. Simpson lsning i middle?

KUNNE DET VRE INTERESSANT AT BRUGE DYNAMIC PROGRAMMING? KUNNE MAN GEMME HVILKE KALD TIL INNER DER ER LAVET OG HVIS INNER S KALDES IGEN MED SAMME VRDIER, KAN MAN SLIPPE FOR AT REGNE DET UD OG BARE HENTE RESULTATET. DOG SKAL MAN VRE OPMRKSOM P AT MAN KAN KOMME I EN SITUATION HVOR EN BEREGNING SER AT DER IKKE ER NOGET RESULTAT FOR ET ST PARAMETRE OG DERFOR GR I GANG MED AT REGNE DET UD, MENS DEN FRST BEREGNING KRER KAN ANDRE BEREGNINGER MED SAMME PARAMETRE OG DE NYE BEREGNINGER VIL OGS GERNE GEMME RESULTATET NR DE ER FRDIGE HVILKET KAN GIVE CONFLICTS (MSKE). eller Det kan vre vi skal dokumentere en del om at man kan gemme alle resultater i en tabel og efter et vist antal kunder kan man ende med nok til bare at have opslag i stedet for beregninger. skal denne tabel et ligge p GPU s den kan sls op i konstant tid.

skriv evt noget om autotuning, hvor programmet frst tester med forskellige mngder af trde og blokke og vlger den hurtigste lsning.

- Did we achieve what we wanted? what did we discover during the project? What can be changed in future implementations? Threats to validity??? kan det betale sig at kre dele af programmet p CPU'en?

11 Glossary

- Forsikringstager - insurance holder
- gtefille - Spouse
- livrente - life interest
- ddfaldssum
- pause periode - grace period
- randbetingelse - boundary condition

References

- [1] EDLUND DOKUMENTET
- [2] David B. Kirk, Wen-mei W. Hwu - Programming Massively Parallel Processors, A Hands-on Approach - Elsevier Inc. - 2010