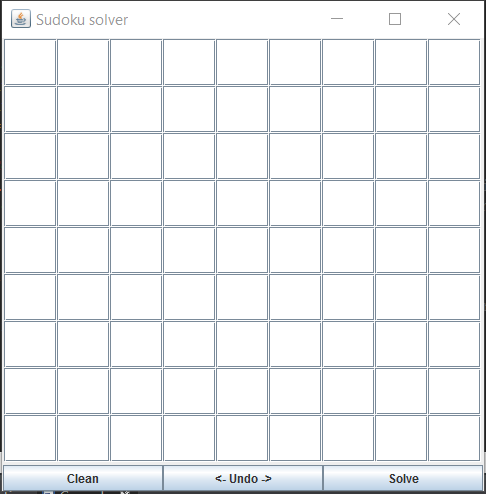
Draft Programming Expert Project :

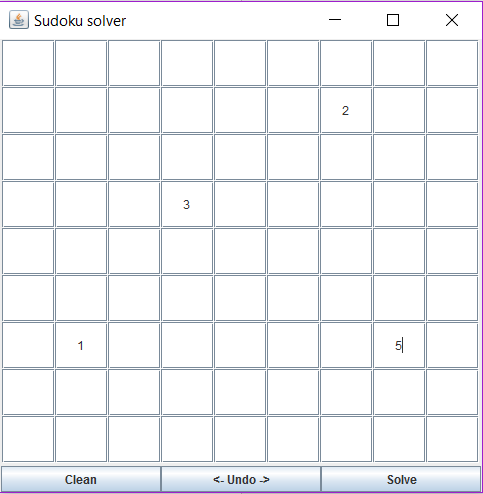
Sudoku Solver

# Werking

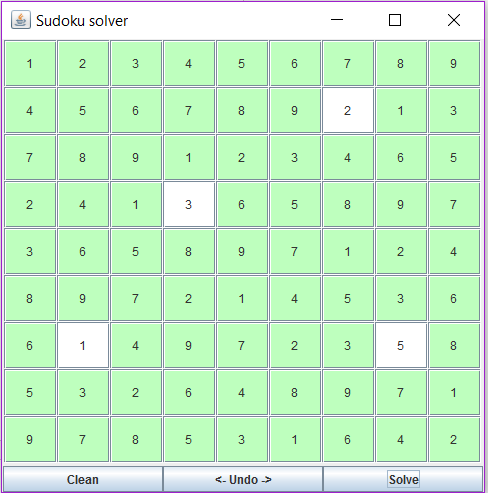
Dit is scherm dat men krijgt bij het opstarten van het programma. Hier heb je drie verschillende functies. De solve functie, waarbij de sudoku wordt opgelost. De undo functie, waarbij de vorige actie ongedaan wordt. De clean functie, waarbij de sudoku wordt gereset.



Als volgende moeten de getallen ingegeven worden. Je kan zelf kiezen welke getallen je waar gaat plaatsen.



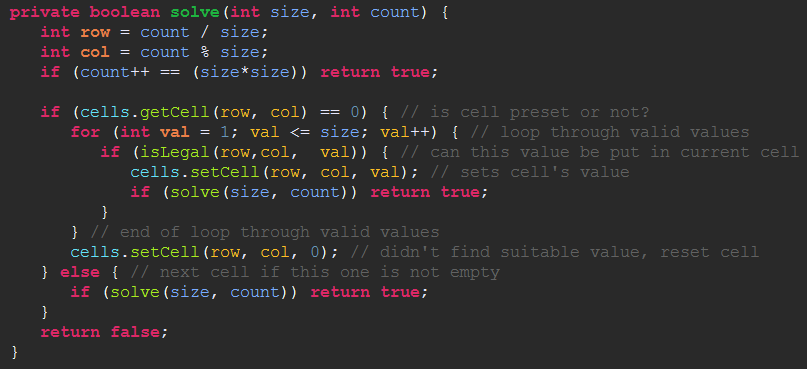
Wanneer je de getallen ingevoerd hebt en op “solve” klikt gebeurt er dit.



Als extra functie kan je de suduko resetten door op clean te klikken, zodat je constant aan een nieuwe sudoku kan beginnen.

# Algoritme

Bij dit project wordt gebruik gemaakt van het backtrack algoritme. Hierbij moeten niet alle oplossingen bekeken worden. Bij dit algoritme zal het programma een aantal keuzes moeten maken. Wanneer het programma op een dood spoor terecht komt, zal het programma terug naar het begin gaan en andere keuzes proberen te zoeken die wel werken. Dus bij deze sudoku solver zal hij nagaan of hij deze waarde in een bepaalde cel kan gooien. Indien niet, dan zal hij verder kijken voor een andere waarde. Zo zal hij die lus blijven afgaan tot als hij een oplossing gevonden heeft. Hieronder is de algemene methode die deze sudoku solver gebruikt:



Referentie naar artikel over het backtrack algoritme:

<http://algorithms.tutorialhorizon.com/introduction-to-backtracking-programming/>

In bijlage vindt u ook een uitgebreide PowerPoint waar ze dieper ingaan op het algoritme.

# Tools

## JConsole

We hebben voor onze meetresultaten gebruik gemaakt van JConsole. Dit is een interessante tools om performantieanalyses te doen. Dit geeft een mooi overzicht van de gebruikte memory van de applicatie en de threads die er momenteel gaande zijn. In de meetresultaten vindt u enkele screenshots terug van JConsole in zijn werking.

Referentie naar JConsole documentation:

<http://docs.oracle.com/javase/7/docs/technotes/guides/management/jconsole.html>

## VisualVM

## JProfiler

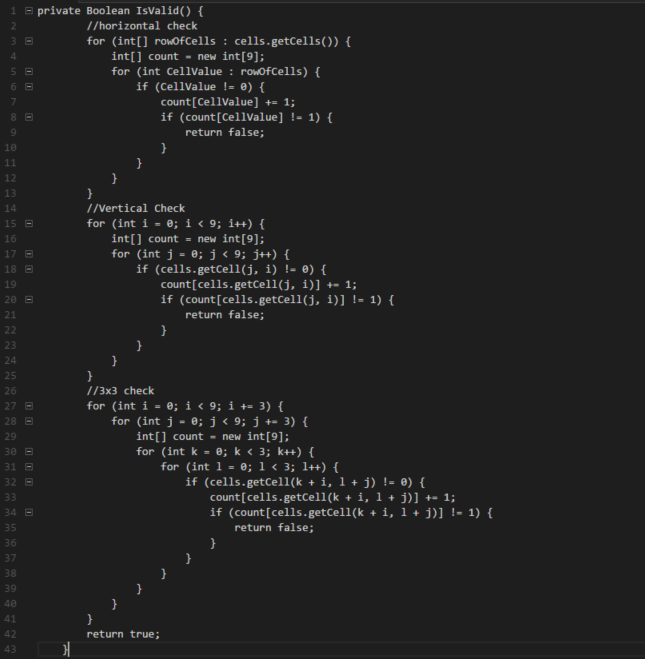
# Analyse

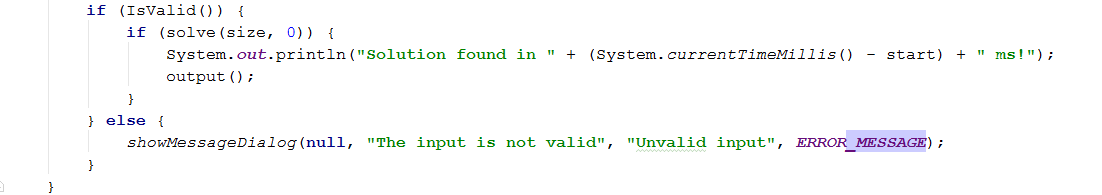
## Probleem #1: Crash programma bij foute invoer

### Beschrijving

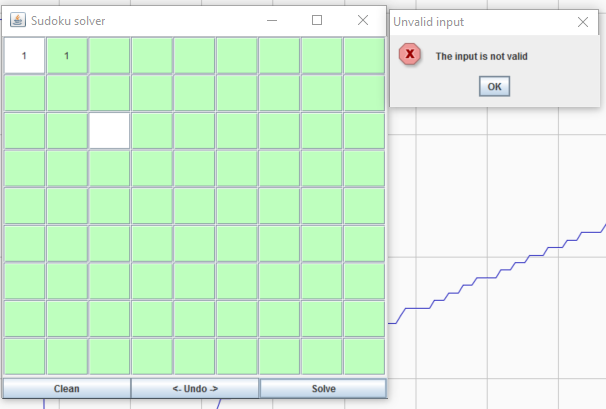
Als het programma een sudoku krijgt gegeven die onmogelijk op te lossen is, blijft het vaststeken voor 3-5 minuten omdat het programma het toch probeert op te lossen.  
Hiervoor zou een controle geschreven kunnen worden zodat dit niet gebeurt maar rechtstreeks een error word gegeven.

### Oplossing





Door hier dan gebruik te maken van een simpele if test, zorgen we ervoor dat deze fout ten alle tijden wordt opgevangen.



Nu word er rechtstreeks een foutmelding gegeven binnen de 3ms dat deze sudoku niet opgelost kan worden in de plaats van 5 minuten zonder een foutmelding.

## Probleem #2: Stijgende memory

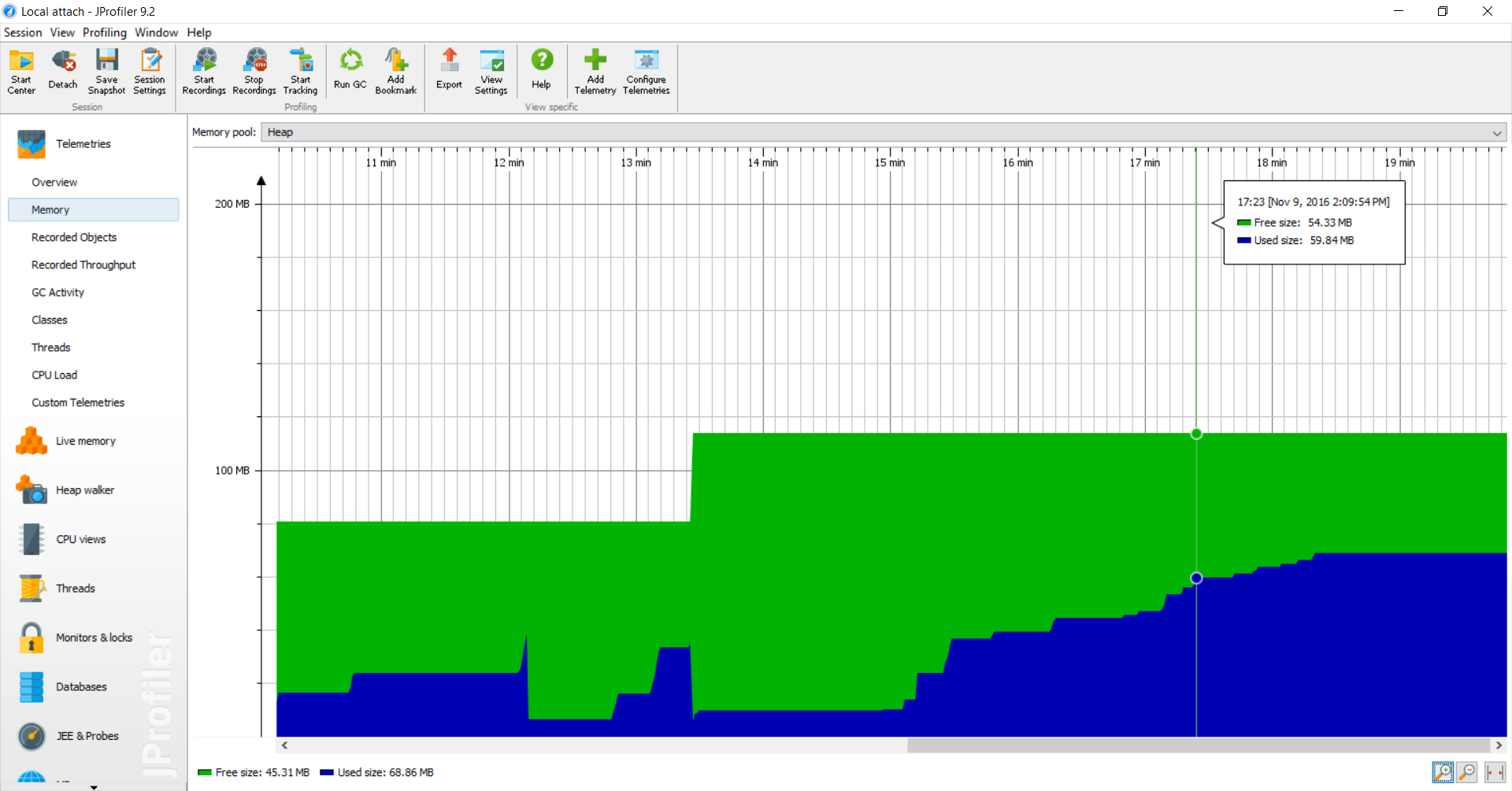
### Beschrijving

We hebben gemerkt dat als we een memory analyse deden op het programma dat de memory hoger en hoger werd. Soms als de memory een piek bereikt wordt de heap verhoogd, zodat er de volgende keer meer memory gebruikt kan worden. Het is onnodig dat dit programma veel mermory gebruikt, dus zochten wij hier een oplossing voor. Dit komt voor wanneer er telkens op “solve” wordt geklikt.

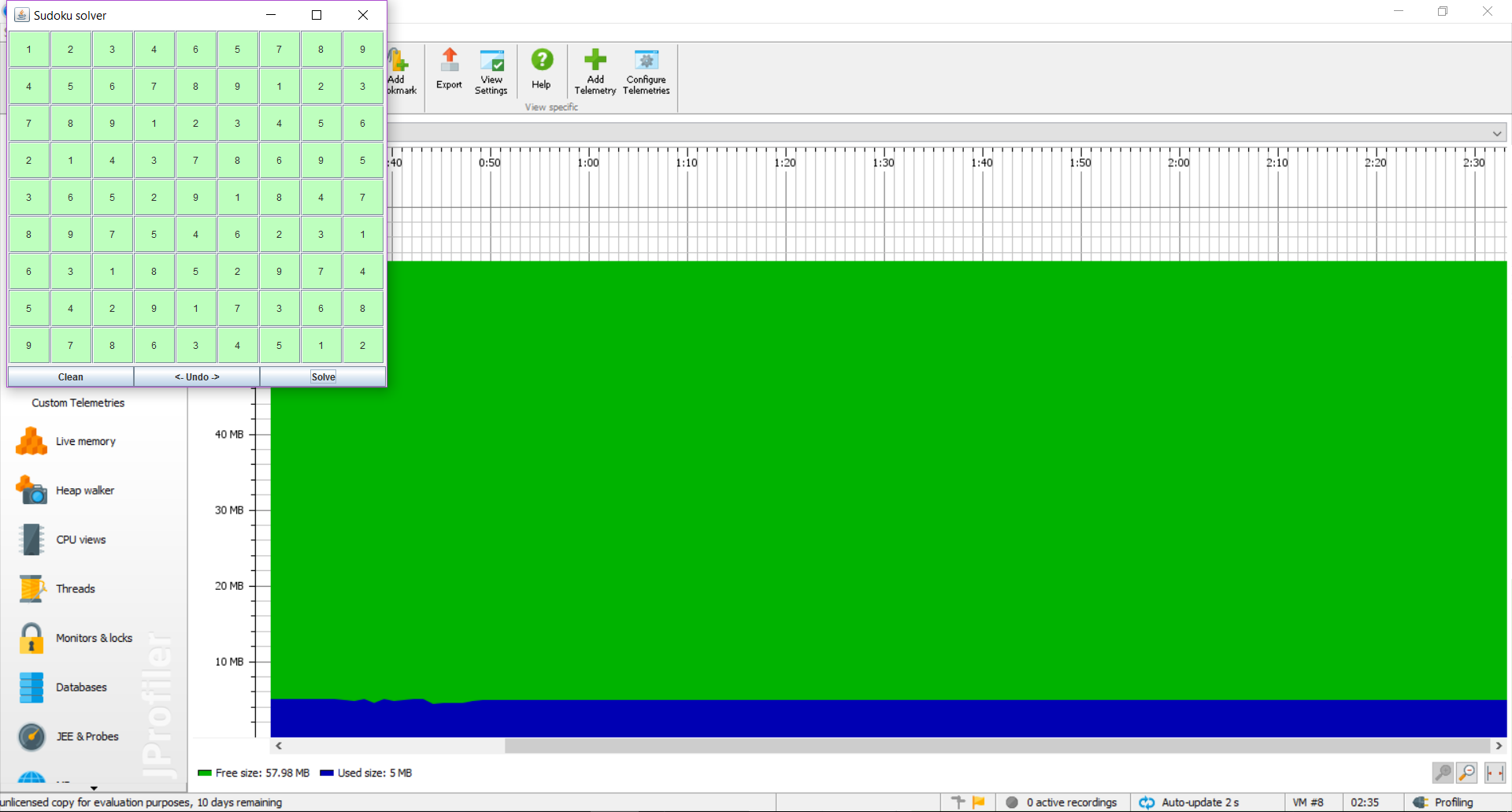
### Meetresultaten

Zoals u hieronder kan zien, start het met een maximum heap size van onder 100mb. Wanneer we op solve blijven klikken en de memory in de lucht blijft gaan, merken we op dat de maximum heap size verhoogd wordt boven de 100mb. Daarnaast merken we ook op wanneer de heap size verhoogd wordt dat de memory ook hoger kan gaan dan normaal.

### 



Oplossing



## Probleem #3:

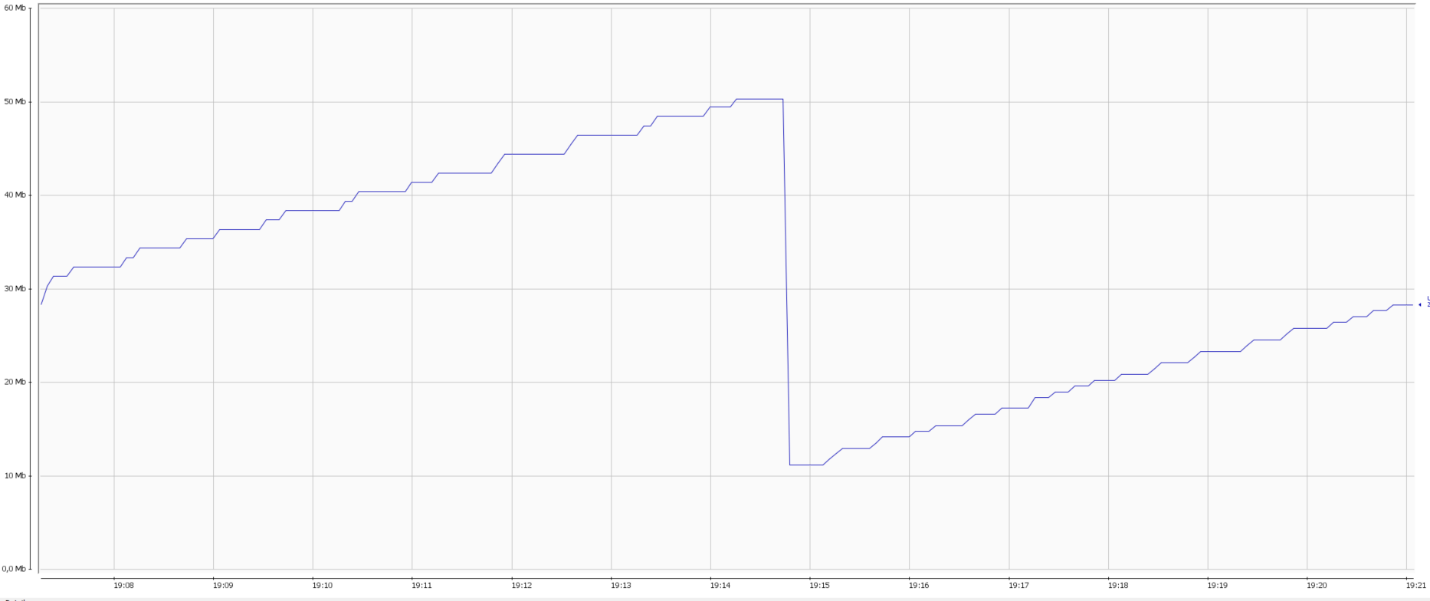
### Beschrijving

### Meetresultaten

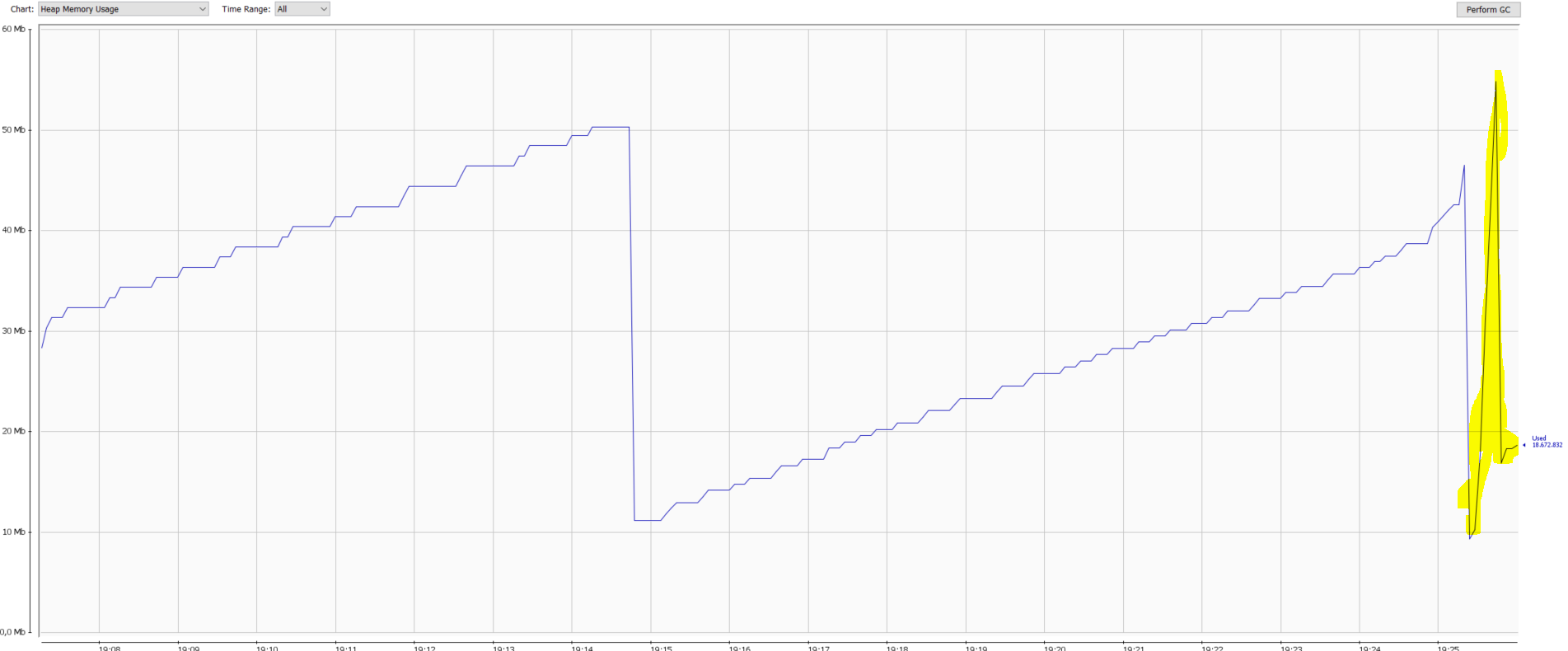
### Oplossing

# Meetresultaten

Als het programma wordt opgestart gebruikt het geleidelijk aan meer geheugen, ook al word er niks mee gedaan. Ik heb niet gevonden hoe dit komt omdat het programma eigenlijk niks doet. Maar ik vermoed dat dit de ui is.



Als ik op de undo button blijf klikken schiet het geheugen omhoog maar deze wordt ook weer opgenomen over de 50mb. Bij het oplossen van de sudoku gebeurt hetzelfde. Als er op clear word gedrukt word de ui wel leeggemaakt maar de waarden worden in een andere variable gezet zodat de undo button nog steeds werkt. De undo button werkt ook nog wanneer de garbage collection gebeurt is.



2) De ui gebruikt telkens meer memory zonder dat er in het programma wat gedaan wordt. De ui zou anders opgebouwd kunnen worden zodat dit niet gebeurt.

3) Het oplossen van de sudoku gebeurt meestal binnen de 3-5ms. Dit zou versneld kunnen worden door gebruik te maken van multithreading. Hierdoor zouden ook meerdere oplossingen kunnen terug gegeven worden want nu krijg je altijd maar één oplossing terug terwijl er veel meer mogelijk zijn.

