Intervallträd som grund vid tillgänglighetsanalys

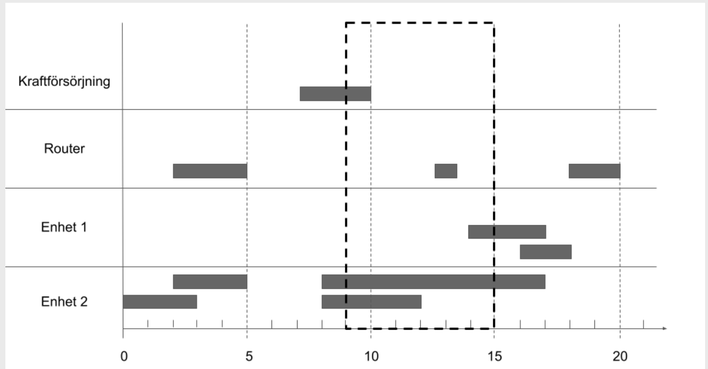
TRAFIKVERKET

FÖRFATTARE

Marcus Sjökvist.

INTRODUKTION

Det svenska transportsystemets infrastruktur kräver många olika tekniska enheter av diverse olika typer för att fungera såsom exempelvis sensorer, kameror, digitala skärmar och routrar. För att säkerställa att de når upp till en god standard så finns det ett *Service Level Agreement* (SLA) som ställer krav på driftsäkerheten. För att kunna effektivt följa upp på SLA och upptäcka brister hos tjänster och utrustningar krävs en effektiv algoritm för att söka bland data och beräkna tillgänglighetsmåtten utifrån dessa. Detta examensarbete har undersökt, utifrån tillhandahållen testdata, huruvida detta kan åstadkommas med hjälp av datastrukturen intervallträd. För att undersöka detta byggdes en prototyp i form av en API-applikation som söker och filtrerar bland hundratusentals larmdata och beräknar tillgänglighet utifrån olika mått såsom procentuell upptid, medeltid till fel och medeltid till återhämtning.



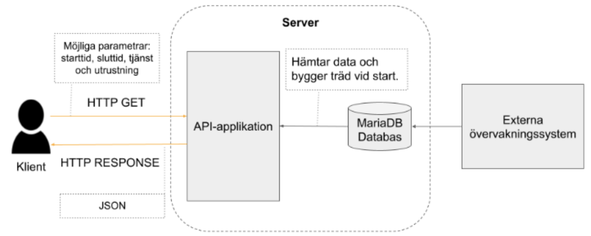
Exempel på tjänst med flera enheter att analysera. X-axeln visar tid. Rektangel med streckad linje visar exempel på en tidsram [9, 15] att söka efter genomskärande intervall för att sammanställa upptid. Larm som nertid visas som grå horisontella rektanglar.

METOD

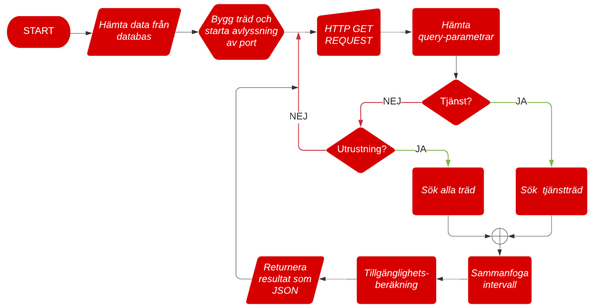
Examensarbetet indelades i fyra olika faser: planering, förstudier, programmering och testning. Planeringen bestod av diskussioner med Trafikverket för att kartlägga problemområdet. Efter denna kartläggning kunde tidsuppfattningar göras och därmed även avgränsningar. Förstudierna bestod till en början av internetsökningar i allmänorienterande syfte och därefter studerades böcker om datastrukturer och algoritmer samt olika vetenskapliga artiklar för att kunna skapa en planerad lösning. Denna lösning implementerades sedan under programmeringsfasen. En alternativ och enklare lösning som inte använder sig av intervallträd implementerades även som jämförelseobjekt. Den sista testfasen bestod av tester av sökning, med både olika parametrar och olika mängder sökträffar, samt uppstartstid. Alla tester utfördes tre gånger och sammanställdes som genomsnittsvärden.

LÖSNING

Det färdiga systemet är ett system bestående av en API-applikation i JavaScript, med Node.js, och en databas. Applikationen bygger upp intervallträden baserat på tjänster vid uppstart genom att hämtad data från databasen. Användaren kan kommunicera med API:et genom HTTP-metoden GET där parametrarna starts, sluttid, tjänst och utrustning är möjliga. Alla parametrarna är valfria men tjänst eller utrustning måste anges. För att intervallträdet ska vara effektivt i sökningarna behövs trädet byggas på förhand då sökning sker på O(log n + k) medan uppbyggnad av ett intervallträd sker under O(n log n), där n är antalet noder.



Planerad applikationsarkitektur. I den slutgiltiga prototypen plaserades API-applikationen på en lokal server.



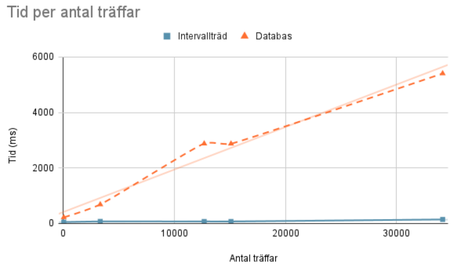
Flowchart för prototypen.

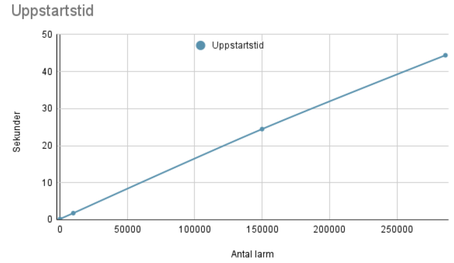
Inuti applikation används intervallträdet för att strukturera data och söka efter överlappande intervall. Därefter sammanfogas dessa intervall med hjälp av datastrukturen stack. Därefter beräknas olika tillgänglighetsmått såsom procentuell upptid, MTTR, MTBF total nertid och total upptid och returneras som JSON-objekt.

Den alternativa prototypen är lik intervallträdslösningen förutom i det avseendet om hur och när överlappande intervall hämtas. I denna variant hämtas intervallen från databasen genom en SQL-sats då en förfrågan till API:et inkommer istället för att de hämtas från de redan konstruerade intervallträden.

RESULTAT

Resultaten av testerna visar att den intervallträdsbaserade prototypen är ungefär två till fyrtio gånger snabbare vid sökning. Två gånger snabbare vid fallet av en singel sökträff och fyrtio gånger snabbare vid tiotusentals sökträffar. Denna prestanda kommer till priset av en uppstartstid. Testerna visar en relativt linjär relation mellan antal träffar och uppstartstiden där ungefär 300 000 träffar leder till en uppstartstid på i genomsnitt 44 sekunder.





DISKUSSION

Detta examensarbete har kunnat påvisa att intervallträdet och den framtagna algoritmen för tillgänglighetsberäkning är en möjlig lösning för det problemområde som undersökts. Medan resultaten visar att bägge lösningar fungerar så visar de även att intervallträdssökningen till och med presterar bättre i samtliga tester där den som sämst har presterat ungefär 208% och som bäst ungefär 4 100% av den naiva lösningens effektivitet tidsmässigt. Resultaten visar även på att lösningen är mycket mer skalbar än den alternativa lösningen. Den nackdel som påträffats med lösningen är att den har en betydande uppstartstid. Detta behövs dock bara göras under underhåll och är på så vis överkomlig.

Vid en eventuell produktionssättning skulle autentiseringsfunktioner behöva tillkomma samt bör HTTP-metoden POST användas istället för GET. Det skulle även behövas ett skript som uppdaterar intervallträden när nya larm tillkommer i databasen.

Prototypen är gjord specifikt för Trafikverkets IT-miljö men skulle även med få anpassningar kunna användas inom alla olika sorters IT-system så länge det finns intervalldata tillgänglig.