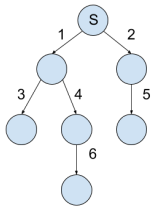
EXAMENSARBETE Implementing and Evaluating a Breadth-First Search in Cypher
STUDENTER Alexander Olsson, Therese Magnusson
HANDLEDARE Krzysztof Kuchcinski (LTH), Tobias Lindaaker (Neo4j)
EXAMINATOR Flavius Gruian (LTH)

Kan vi söka effektivare i grafdatabaser?

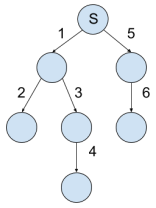
POPULÄRVETENSKAPLIG SAMMANFATTNING **Alexander Olsson, Therese Magnusson**

Grafdatabaser blir mer och mer populära. När popularitet ökar så önskas det att sökningar i grafdatabasen utförs snabbt och korrekt. Därför behövs det snabba och pålitliga sökalgoritmer för att genomföra sökningarna.

Eftersom grafdatabaser blir mer och mer populära behövs bra sökalgoritmer för att snabbt och pålitligt söka efter matchningar till den givna förfrågan. De två vanligaste sökalgoritmerna i grafer är bredden-först sökning (BFS) och djupet-först sökning (DFS). BFS och DFS är två liknande algoritmer, där skillnaden är att BFS söker brett först medans DFS söker djupt först. Exempel på deras sökordningar visas i figurerna nedan.



En BFS sökning



En DFS sökning

Båda algoritmerna har länge studerats men ej i anslutning till grafdatabaser. Grafdatabasen Neo4j och dess språk Cypher kan få mycket nytta med att ha en BFS operator utöver deras nuvarande DFS operator. Detta eftersom BFS och DFS har olika styrkor och vilken av dem som passar bäst för ett visst problem är ej självklart innan de jämförs på, iallafall, liknande problem.

Grafdatabaser är en form av databas som lagrar sin information i grafer, där relationerna mellan datan sparas. Exempel på områden där detta är användbart är bedrägeri detektering, nätverk- och IT-verksamhet, identitets- och åtkomsthantering och motorer för realtidsrekommendationer[1]. Grafdatabaser används bland annat av Google, Facebook, LinkedIn och PayPal.

I vårt examensarbete samarbetade vi med företaget Neo4j och utvecklade en BFS prototyp med flera olika optimeringar till deras grafdatabas. De utvärderades gentemot varandra och den redan existerande sökalgoritmen. De optimeringar vi implementerade rörde att dela upp sökningen i flera mindre sökningar, så kallat Top down Bottom up mönster. Fördelen med dessa optimeringar är att undvika onödiga noder i sökningen.

Resultatet från vår utvärdering visade att våran BFS prototyp vanligen var jämlik med DFS operatorn. Men för mindre grafer var vi upp till 90% snabbare och för större grafer upp till sex gånger långsammare. Överlag var BFS bäst för mindre grafer och mindre sökningar, med strängare restriktioner. Optimeringarna gav signifikanta förbättringar för större grafer. De kräver dock ännu fler restriktioner och är därför svårare att använda i dagliga situationer.

Vår utvärdering visade att använda BFS på grafdatabaser kan ge både stora förbättringar och försämringar, beroende på grafen. Detta är intressant eftersom det visar att detta är ett område som borde utforskas mer.

[1] https://neo4j.com/why-graph-databases/