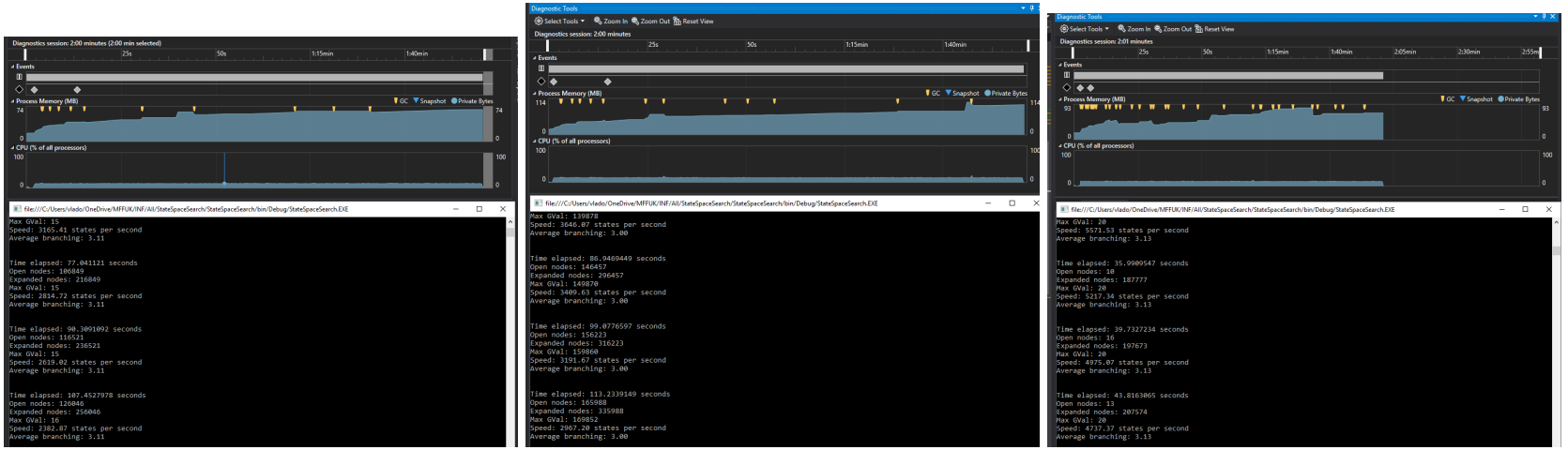
3. Vstup Fifteen Puzzle



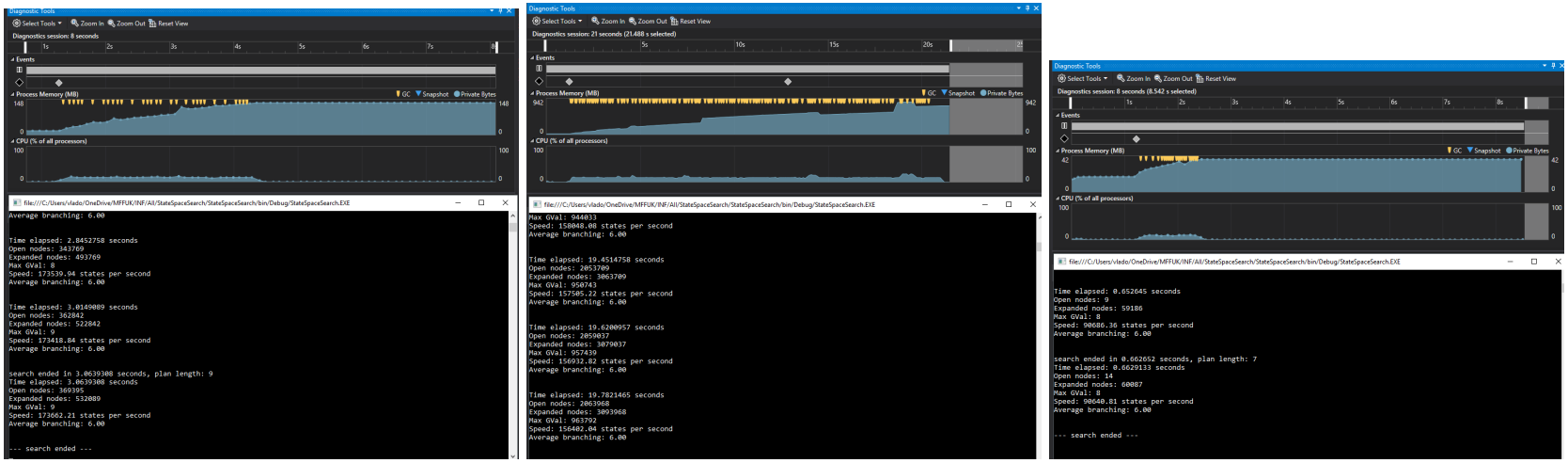
Na tomto vstupe vidno chovanie jednotlivych algoritmov:

Breadth First Search (BFS, vlavo) prechadza vzdy vsetky vrcholy na hladine. Na to ich ale uklada, co vedie na velky pocet otvorenych vrcholov (a potencialne zbytocne velke mnozstvo spotrebovanej pamate)

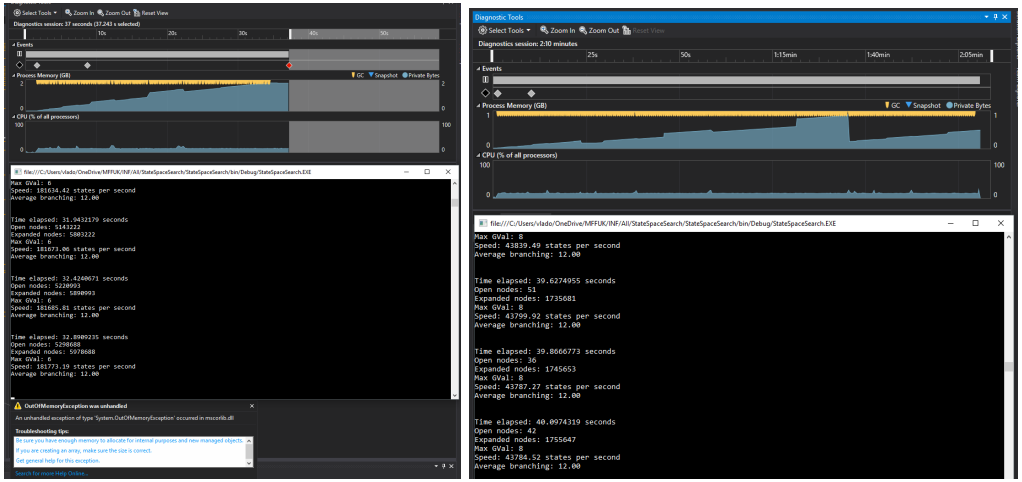
Depth First Search (DFS, stred) prechadza prvotne vrcholy do hlbky. Vyhodou je, ze najde hlboko skryte riesenie extremne rychlo (ak sa mu to podari, neskor uvidime priklad), ale na druhej strane sa algoritmus moze zbytocne dostat do hlbky, ktoru v rieseni skumat vobec netreba.

Iterative Deepening Search (IDS, vpravo) je nieco medzi - pocet otvorenych vrcholov je podobny DFS, kym hlbka, do ktorej sa algoritmus dostava pripomina BFS. Vsimnime si tiez graf vyuzitej pamate, ktory je menej "hladky". Toto chovanie je sposobene jednotlivymi iteraciami behu algoritmu, kedy sa halda vzdy uplne vyprazdni a zacne naplnat znovu.

2. Vstup Rubikovej kocky



3. Vstup Rubikovej kocky



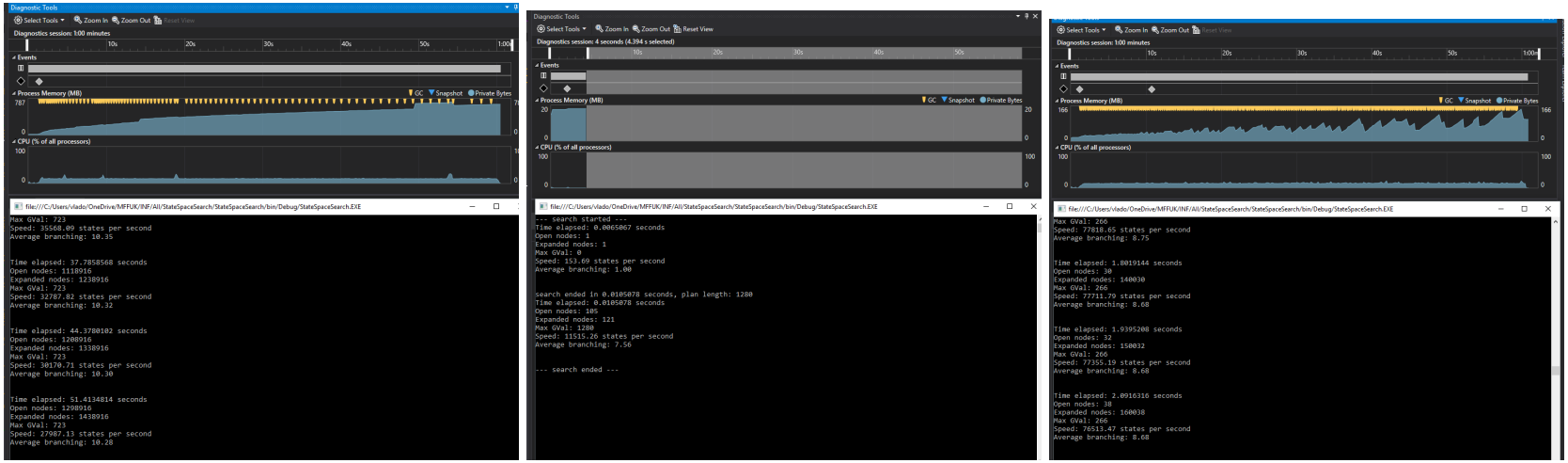
Tieto vstupy su prikladom, kedy DFS zbytocne nabera hlbku a u BFS dochadza pamat.

BFS (vlavo) Najde riesenie, ale zaberie pomerne dost pamate. Na 3. vstupe to zacina byt vazny problem, kde algoritmus skonci na *OutOfMemoryException*.

DFS (stred) Ide v tomto pripade zbytocne hlboko - hladat riesenie rubikovej kocky v milionovych hlbkach je jednoducho zbytocne.

IDS (vpravo) Najde riesenie a na viac zaberie len zlomok pamate BFS. Toto je vyhodou na 3. vstupe, kde algoritmu dochadza pamat pomalsie.

2. Vstup Obchodneho cestujuceho



Na tomto vstupe vidno priklad, kedy riesenie vratene DFS nie je odpovedou na zadanu ulohu:

BFS (vlavo) Postuple prechadza vsetky hladiny a preto je uloha "najst nejake riesenie" rovno ulohou "najst najlepsie riesenie". V tomo pripade to ale bohuzial znamena, ze pocet otvorenych vrcholov (a teda aj mnozstvo potrebnej pamate) rychlo stupa.

DFS (stred) V tomto pripade na nejake riesenie narazi pomerne rychlo - algoritmus vrati 1. cestu, ktora obsahuje vsetky vrcholy grafu, bez ohladu na jej dlzku (t.j. riesenie nie je optimalne)

IDS (vpravo) Teraz robi to iste, co BFS (postupne skuma hladiny), t.j. najde najlepsie riesenie - vyhodou ale je, ze si nemusime drzat zoznam vrcholov na hladine. Na grafe vyuzitia pamate pekne vidno jednotlive iteracie behu algoritmu.