6.7.4 Computation Complexity

Introduction: The *Computation Complexity* one mission control run assessment is necessary to identify the strong and weak points of approach. Lets get through modules to assess notable calculations/algorithms complexity on high abstraction level.

Navigation Loop: I the navigation loop, the *waypoint reach condition* (eq. **??**) is checked, this is unitary operation with worst complexity $\mathcal{O}(1)$. The selection process of the next *Goal Waypoint* can get through all waypoints in the mission if they are all unreachable the complexity is $\mathcal{O}(|waypoints|)$.

The *notable steps* complexity is following:

$$\begin{aligned} \texttt{Reach Condition:} &\quad \mathcal{O}(1) \\ \texttt{Select Next Waypoint:} &\quad \mathcal{O}(|waypoints|) \end{aligned}$$

Data Fusion: The *data fusion* is all about *threat selection.*

If *UAS* is in *controlled airspace* it needs to iterate over received *collision Cases* to select *active ones.* The complexity of this step is linear, therefore boundary is given as $\mathcal{O}(|collisionCases|)$.

Thresholding *Detected Obstacles* is done by simple comparison of *LiDAR ray hits* in given $cell_{i,j,k}$ of *Avoidance Grid.*

Any loading of *threats* from *information sources* is depending on clustering. The *Airspace Clustering* is considered as static for our setup. Therefore the *count of active airspace clusters* has main impact on complexity. The *count of information sources* is static and not changing over mission time. Information sources usually implement *Hash search function* with complexity $\mathcal{O}\ln|searchedItemSet|$.
The *computation complexity* boundaries for *Data fusion* in our setup are following:

$$\begin{aligned} \texttt{Select Active Collision Cases:} &\quad \mathcal{O}(|collisionCases|) \\ \texttt{Threshold Detected Obstacles:} &\quad \mathcal{O}(|cells|) \\ \texttt{Load Map Obstacles:} &\quad \mathcal{O}(\ln|activeClusters| \times |informationSources|) \\ \texttt{Load Hard Constraints:} &\quad \mathcal{O}(\ln|activeClusters| \times |informationSources|) \\ \texttt{Load Soft Constraints:} &\quad \mathcal{O}(\ln|activeClusters| \times |informationSources|) \end{aligned}$$

Note. The *real-time clustering* is *hard non-polynomial problem* [1]. Usually all information sources and sensor have *polynomial complexity* of processing. The *controlled airspace clusters* are usually set for very long period of time. Therefore *Obstacle Map*, *Airspace Constraints*, and, *Weather Constraints* can be considered as preprocessed

Situation Assessment: The *Situation Assessment* is evaluating triggering events. The *evaluation* is usually simple existence question without further calculations. The *complexity* of *event evaluation* for our case is $\mathcal{O}(1)$. There are 8 triggers. The count of *triggers* needs to be accounted in complexity boundary:

$$\mathcal{O}(|triggers| \times eventEvaluationComplexity)$$

Note. The *trigger calculation complexity* needs to stay low, because the *triggers* are verified every *Mission Control Run.* The *Avoidance Run* trigger frequency should be very low under normal conditions.

Avoidance Run: The *Avoidance run* is most critical part of *Mission Control Run*, because *Avoidance Path* calculation. The *Navigation Path* calculation is less complex (Rule engine is not accounted), therefore *Emergency Avoidance Mode* is assumed.

The *threat insertion* is realized in 7^{th} to 10^{th} step. The first is *Avoidance Grid* filled with *Static Obstacles.* The *Avoidance Grid* is designed to separate rotary *LiDAR* ray space into hit count even cells. Insertion of *LiDAR* scan into *Avoidance Grid* complexity depends on *total cell count.* The *upper boundary* for *insert obstacles* is given like follow:

$$\texttt{Insert Obstacles:} \quad \mathcal{O}(|cells|)$$

The *intruders intersection model* type impact the insertion complexity. The *linear intersection* (sec. **??**) is going through maximum of *layers count* cells.

The *body volume intersection model* (sec. **??**) can check the *simple intersection condition* over all *Avoidance Grid* in worst case, therefore complexity for this check is bounded by *count of cells.*

The *Maneuverability Uncertainty Intersection* (sec. **??**) can hit all cells in *Avoidance Grid.* The calculation complexity boundary is exponential depending on *horizontal/vertical* spread in $[rad]$. The *intersection* implementation was done *ad-hoc.* The impact of *intersection application* is visible only when there is more than *4* concurrence intruders (fig. **??**).
The *complexity boundary for* intruder insertion is given like follow:

$$\texttt{Insert Intruders:} \quad \mathcal{O}\left(\sum \begin{bmatrix} |linearIntersections| \times |layers| \\ |bodyvolumeIntersections| \times |cells| \\ |cells|^{horizontalSpread \times verticalSpread} \end{bmatrix}\right)$$

Note. The *intruder intersection* is critical in *non-controlled airspace.* The main complexity gain in *controlled airspace* is from *rule application.* Our *rule complexity* is in worst case depending on *Reach Set* node count and *Active Collision Cases* count.

$$\texttt{Apply Our Rules:} \quad \mathcal{O}(|activeCollisionCases| \times |nodes|)$$

For *Hard/Soft Constraints* The algorithm used for intersection polygons was selected based on study [2], the selected algorithm *Shamos-Hoey* [3]. The *calculation complexity* boundary is given like follow:

$$\texttt{Hard Constraints Intersection:}\quad \mathcal{O}(|cells| \times |hardConstraints| \times \max|constraintPoints|^2)$$

$$\texttt{Soft Constraints Intersection:}\quad \mathcal{O}(|cells| \times |softConstraints| \times \max|constraintPoints|^2)$$

Each *threat* category application in *Mission Control Run* is done after *each intersection* in 7^{th} to 10^{th} step. All ratings (tab. **??**) expect $Reachibility(cell_{ij,k})$ and $Reachibility(Trajectory)$ are calculated. The *calculation complexity* boundary for one *reachibility rating* is $\mathcal{O}(1)$. (eq. **??**, **??**). The *Recalculate Reachibility* operation applied $4\times$ have maximal *complexity* boundary given as follow:

$$\texttt{Recalculate Reachibility:}\quad \mathcal{O}(4 \times (|nodes| + |cells|))$$

Each time at the end of in 7^{th} to 10^{th} step the *Avoidance Path is Selected*. The *Worst Case* (expected) scenario is to *select* four paths for each *treath* application. The algorithm for *best path selection* (alg. **??**) iterates over all *cells* in avoidance grid and over all *trajectories* passing through that cell. The complexity boundary for *path selection* is given as follow:

$$\texttt{Select Path:}\quad \mathcal{O}\left(4 \times \left(|cells| + \frac{|nodes|}{|cells|}\right)\right)$$

Conclusion: Overall approach complexity is *low*. If proper *Information Sources* with efficient clustering and *intersection models for intruders* are used, the approach will stay within *non-polynomial complexity*. The average load time for *testing scenarios* is summarized in (tab. 7.1).

Note. The calculation of *Reach Set* is eliminated by pre-calculation for *state range* [4].

7.5.3 Computation Footprint

The *computation footprint* is summarized in computation load (tab. 7.1). The *computation load* (eq. ??) was calculated for each *time-frame* in scenarios. There is summary of *minimal, maximal, average* and *median* values.

The *computational load* never exceed more than 55.95% in case of *emergency Head On* (eq. ??), which means that *every path* was calculated on time.

Scenario	Computation load			
	min.	max.	avg.	med.
Building avoidance (fig. ??)	2.20%	27.40%	12.11%	13.20%
Slalom (fig. ??)	12.20%	30.50%	21.42%	21.50%
Maze (fig. ??)	24.90%	46.10%	31.51%	30.80%
Storm (fig. ??)	2.60%	26.90%	11.57%	13.90%
Emergency Converging (fig. ??)	2.75%	16.50%	5.84%	4.95%
Emergency Head On (fig. ??)	3.90%	55.95%	13.19%	6.90%
Emergency Multiple (fig. ??)	5.90%	52.35%	12.77%	8.56%
Rule-based Converging (fig. ??)	3.60%	13.50%	7.32%	5.97%
Rule-based Head on (fig. ??)	4.65%	41.60%	13.64%	9.30%
Rule-based Multiple (fig. ??)	4.37%	23.30%	11.96%	10.93%
Rule-based Overtake (fig. ??)	3.85%	13.40%	7.62%	6.70%

Table 7.1: *Computation load statistics* for all test cases.

Following observations can be made:

1. *Building avoidance*, *Slalom*, and *Maze* scenarios - the computation load is increasing with the *amount of static obstacles.* The *average load* for *Emergency avoidance mode* in *clustered environment* is 31.51% (Maze).

2. *Storm scenario* - the overall *computation load* is very low due the *moving constraint implementation* (sec. ??).

3. *Emergency Converging/Head On/Multiple* scenarios - the *overall computation load* is quite high due the ineffective *body volume intersection* (sec. ??) implementation.

4. *Rule-based Converging/Head On/Multiple* scenarios - the *median computational* load is low, because of the linear *rule implementation* (sec. ??)

5. *Rule-based Overtake* - the *average computation load* is very low, because only *divergence/convergence* (rule. ??) waypoints are calculated and UAS stays in *navigation mode.*

Bibliography

[1] Jon Kleinberg, Christos Papadimitriou, and Prabhakar Raghavan. A microeconomic view of data mining. *Data mining and knowledge discovery*, 2(4):311–324, 1998.

[2] Jon Louis Bentley and Thomas A Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on computers*, (9):643–647, 1979.

[3] Michael Ian Shamos and Dan Hoey. Geometric intersection problems. In *17th annual symposium on foundations of computer science*, pages 208–215. IEEE, 1976.

[4] Alojz Gomola, João Borges de Sousa, Fernando Lobo Pereira, and Pavel Klang. Obstacle avoidance framework based on reach sets. In *Iberian Robotics conference*, pages 768–779. Springer, 2017.