Design of the Test-Bed for DiVE 3D

The volume of the zones does not change with the number of entities. It is set as constant to 60 m x 60 m x 60 m. The world size remains constant to 2.4 km x 2.4 km x 180 m. So, we have in total 40 x 40 x 3 = 4800 zones. For ensuring minimal bandwidth usage, the research objective is to determine the optimal ratio of AOI volume to zone volume. For testing the causal influence of different ratios on bandwidth usage a set of ratios is chosen for our testing by varying the AOI volume. Please note that this is in contrast to our design choice of testbed for DiVE 2D in which the area of AOI is kept fixed and area of zone is variable.

The rationale behind such design choice is because in DiVE 3d, our objective is to determine the computing needs (bandwidth consumption, CPU and RAM usage) for a single type of movement pattern which reflects the real world usage of the moving entity. For example, we intend to see how to multiple drones being used in cargo delivery task or say pick-and-place task i.e. following a check-point movement pattern can be simulated in our framework. Thus, it makes more sense to change the AOI and observe the causal influence of the number of exchanged entity-to-entity messages and entity-to-zone subscriptions on the computing needs. On the other hand, the research objective for DiVE 2D testbed was to determine the causal influence of three different movement patterns on computing needs; thus, changing zone sizes changed the concentration of entities for each movement pattern and had causal influence on our finding.

Experiments

We have following two research objectives:

1. **Determination of computing needs for ensuring scalability of the number of networked clients:** This is achieved by determination of optimal ratio of volumes of AOI to Zones to ensure minimal bandwidth and CPU usage, for varying number of entities.
2. **Determination of *suitable* alignment of zones for ensuring scalability of the number of networked clients via interest management:** Three modes of Zone alignment are used 1. Y-axis and X-Y axis and 2. No alignment, which is used in Ex 1. The performance on computing needs is compared for among the three modes. Two cases of comparisons are performed:
    a. **General case:** Quantitative evaluation of zone alignment on bandwidth and CPU usage for varying number of entities is tested for every ratio. The three modes are compared.
    b. Specific case: The comparison between three modes is made only for the optimal ratios (calculated as output of Ex 1)

Results

I. **Result of Ex1: Determination of optimal ratio:** The optimal ratio between Area of interest (AOI) and the zone is calculated. There is not distinction between inner and outer area of interest i.e. there is only one AOI. The chosen criteria for determining the optimality is minimization of the bandwidth i.e. the lower it is the more optimal the ratio. The determination of an optimal ratio is important to find an upper bound for the ratio which has stable bandwidth. The measurement of bandwidth is required to determine the communication load on server.

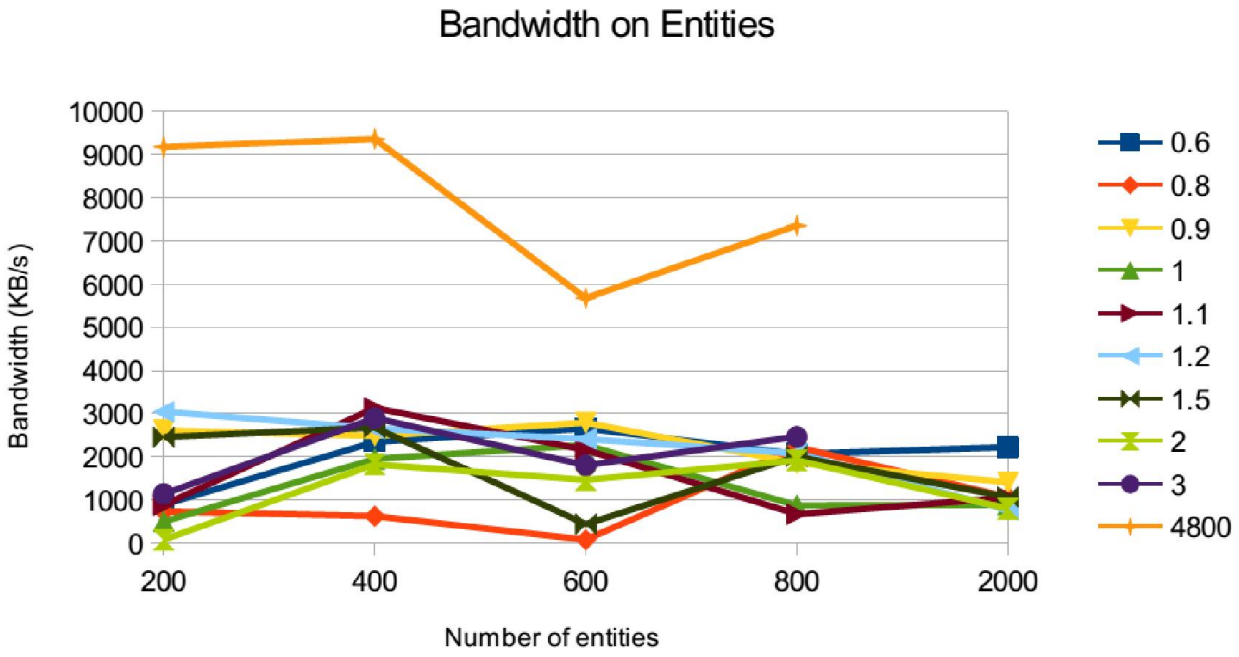

Figure 1: Zones based interest management reduces the bandwidth consumption and ensures scaling compared to the case when there was no interest management (solid line in Orange color). For example, the client crashes when the number of entities are more than 800 and each entity sends messages to other entities (no interest management). For a smaller number of entities i.e. 200, the optimal ratio is 2 which maintains the trade-off between exchange of messages (already limited due to small number of entities) and number of zone subscriptions (less migration from one zone to another). However, if this ratio is kept fixed and number of entities slightly increased then the number of exchanges between the entities will go up which create greater load on the server. Thus, it is preferable to have a smaller ratio (e.g. 0.8) by reducing the AOI volume since this ensures less number of zone subscriptions i.e. lesser exchange of messages.

This however has a drawback of higher migration rate from one zone to another. That is why, a slight increase in ratio is found to be better for a higher number of entities. However, it is observed that when the number of entities become very large then the choice of ratio has no substantial influence on the reduction of bandwidth usage. This can be explained as the tradeoff between the overload on server when two contrasting cases happen i.e. bigger ratio causes higher exchange of messages and smaller ratio causes higher number of zone subscriptions.

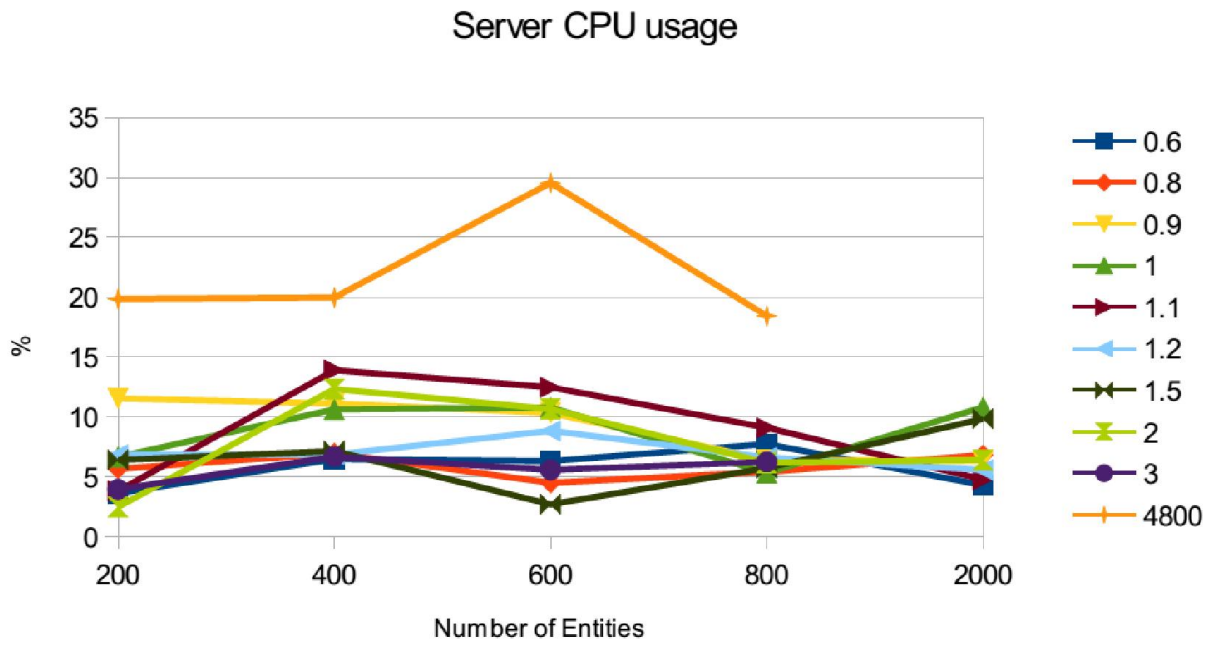

Figure 2: Zone-based interest management reduces the CPU usage with large and small number of entities. With small number of entities i.e. 200, the CPU usage is lower and more convergent comparing to medium and large number of entities. The CPU usage converges when we have large number of entities i.e. 800. In respect to the scalability, without interest management the load of server is too heavy to perform a proper test. With interest management, we could test the entities up to 2000 with stable CPU usage. When we have the medium number of Entities i.e. 600, Rati0 1.5 costs less CPU comparing to other ratio with more obvious difference. Ratio 1.1 costs the most CPU usage when the entities are in medium number, but performs better on small and large number of entities.

Result of Ex2: Determination of *suitable* alignment of zones for ensuring scalability of the number of networked clients via interest management: This experiments contains three settings, No shift, Shifted in Y axis, and Shifted in X and Y axis. The shifted width is 30 m, half of length of the region. Even order rows are shifted. In Shifted in Y axis, we would have one additional zone on shifted rows. As a result we have 4860 zones in total. In shifted in X and Y axis, we have 4901 zones. The more shift being performed, the more marginal regions would be generated. This is the natural short coming of alignment shifting.

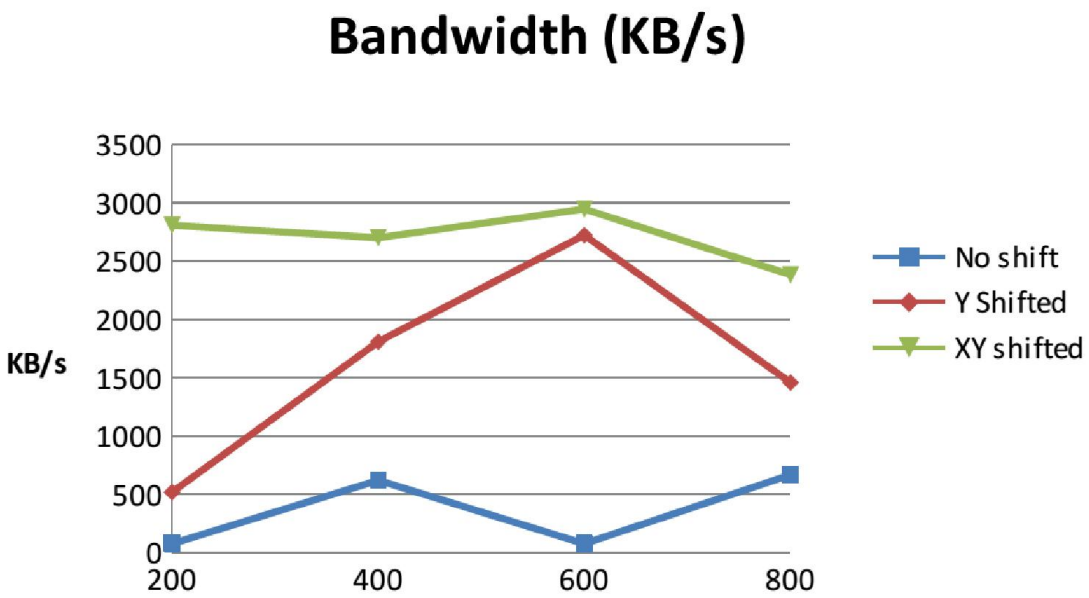

Figure 3. With the optimal ratio we got using the experiment 1, we have four settings with the format (entities, ratio), (200, 2.0), (400, 0.8), (600, 0.8) and (800, 1.1). No shifted cost less bandwidth in all four settings, Shifted in Y axis performs the second best, and Shifted in both X and Y axis have most usage on bandwidth. This comes from the movement pattern that the entities have to move horizontally up and down. In such route, the Shifted contains less zones comparing to No shift. Less region lead to more entities in the same region. Thus a huge amount of bandwidth is generated.

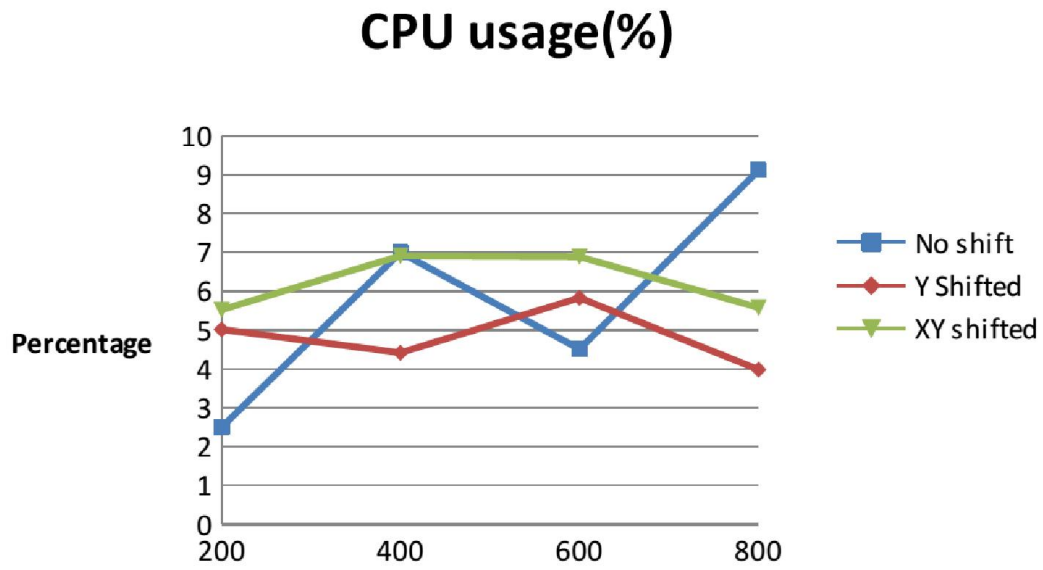

Figure 4. With the optimal ratio we got using the experiment 1, we have four settings with the format (entities, ratio), (200, 2.0), (400, 0.8), (600, 0.8) and (800, 1.1). Shifted in Y generally use less CPU resources comparing to Shifted in X and Y. This effect could be explained by more zones are produced by shifting and thus caused more migration between zones.

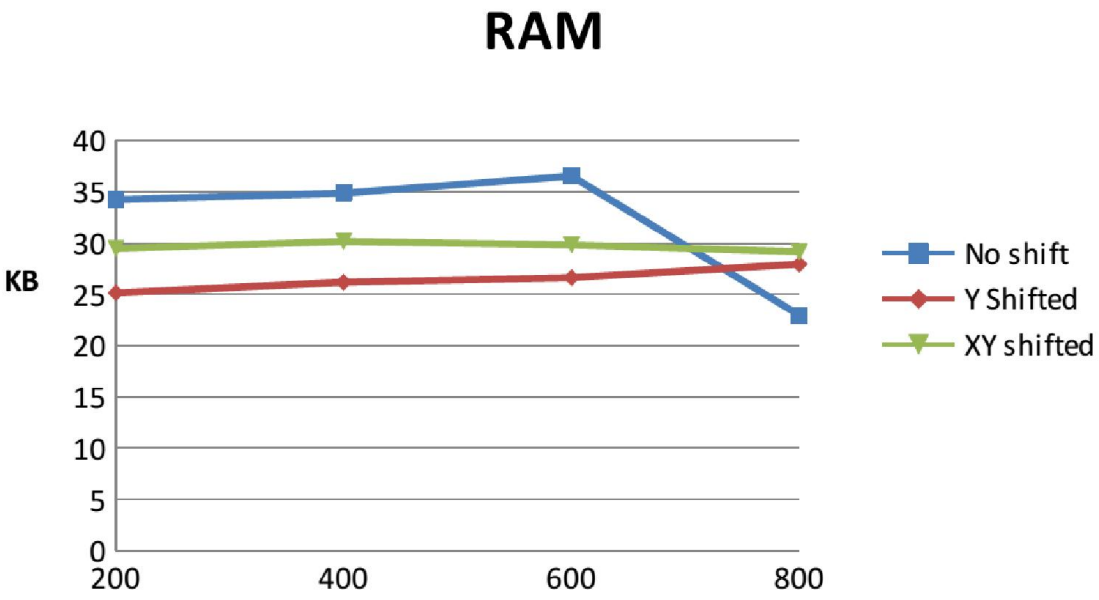

Figure 5. With the optimal ratio we got using the experiment 1, we have four settings with the format (entities, ratio), (200, 2.0), (400, 0.8), (600, 0.8) and (800, 1.1). No shift costs more RAM than others with small amount of entities. However, when the entities goes up, it cost least RAM comparing to others. Shifted in Y costs less RAM comparing to Shifted in X and Y, but the difference shrinks when number of entities goes up.