Assignment #4

Group 4 presents solutions at **2/12**
Group 2 prepares questions/discussion

Sparse matrix – vector multiplication

- Implement the unstructured, sparse matrix – vector multiplication from *Bolz et al. Sparse Matrix Solvers on the GPU: Conjugate Gradients and Multigrid.*
  - You may assume there is exactly two entries per row and no diagonal elements - but do not optimize away R^x.
  - Validate your results

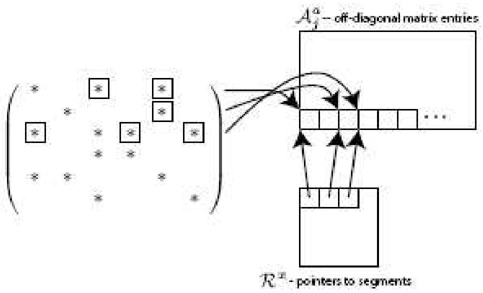

Figure 1: *Off-diagonal elements of each row are compacted into segments which are tightly packed into $\mathcal{A}^a_j$. A pointer to the beginning of each segment is stored in $\mathcal{R}^x$.*

$$
\begin{aligned}
j &= \mathcal{R}^x[i] \\
\mathcal{Y}^x[i] &= \mathcal{A}^x_i[i] * \mathcal{X}^x[i] + \sum_{c=0}^{k_i-1} \mathcal{A}^a_j[j+c] * \mathcal{X}^x[\mathcal{C}^a[j+c]].
\end{aligned}
$$

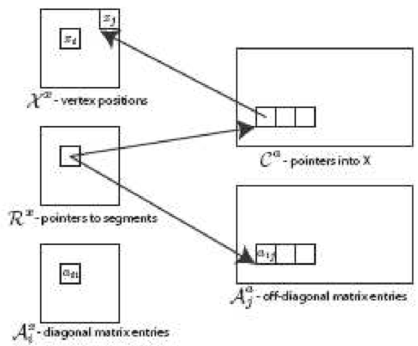

Figure 2: *When the fragment program executes on the pixel corresponding to row i, the window position is used as a texture coordinate to fetch x_i in $\mathcal{X}^x$ and a_{ii} in $\mathcal{A}^x_i$. The window position also identifies the segment pointer in $\mathcal{R}^x$, which points to the location of non-zero elements a_{ij} in $\mathcal{A}^a_j$ corresponding to row i. Finally, using the segment pointer from $\mathcal{R}^x$ we can access entries in $\mathcal{C}^a$ which reveal the addresses of x_j in $\mathcal{X}^x$ corresponding to non-zero a_{ij}.*

Optional, but please try make it

- Implement the structured, sparse matrix – vector multiplication from *Krüger and Westermann. Linear algebra Operators for GPU implementation of Numerical Algorithms*.
  - You may assume there is exactly two bands
  - Validate your results
- Compare the speed of the two methods of matrix-vector multiplication
  - In our algorithm design, does it make sense to aim for either structured or unstructed sparse matrices specifically?