GraphBook: Making Graph Paging Real

Alessio Arleo[1], Felice De Luca[1], Giuseppe Liotta[1], Fabrizio Montecchiani[1], Ioannis G. Tollis[2]

[1] Dip. Di Ingegneria , Università degli Studi di Perugia

[2] University of Crete and Institute of Computer Science-FORTH

Motivation & Idea

In the last years we observed an impressive growth of different mobile devices, like tablets and smartphones, that allow people to quickly access and share a large variety of contents. These devices are often used to access social networks, route networks, or other kinds of networks that for example may deal with the job of the user. Although designing graph drawing algorithms and visualization systems tailored for mobile devices would be the best choice in this case (see, e.g., [1]), this is often not possible, as the drawing might be computed by some diagram server in a collaborative environment, or attached to an e-mail, or returned by a web application that does not take into account limited display capabilities.

We present a system whose goal is to facilitate the reading of a given drawing on a mobile device. To this aim we exploit an analogy with electronic books. It takes as input a drawing Γ of a graph $G=(V,E)$ and computes a **graph book** of Γ. The idea is to adapt the drawing's aspect-ratio to the device display and to distribute the possible visual complexity of the drawing in different pages that can be interactively browsed by using standard gestures. The system works in three main steps. We describe two possible implementations for each step, depending on whether Γ is a straight-line drawing of an undirected graph or an overloaded orthogonal drawing (OOD) [4] of a directed graph. In the first case we can assume that Γ has been computed by some force-directed algorithm (see, e.g., [3]). In the second case we recall that OOD is a recent graph visualization paradigm conceived for directed graphs [4], which has similarities with hierarchical and orthogonal drawings. The presence of an edge (u,v) is conveyed by an *e*-point (a small circle) placed at the intersection point of the vertical segment passing through u and the horizontal segment passing through v. The information in an OOD is enriched by computing the transitive closure of G and by conveying paths as *p*-points.

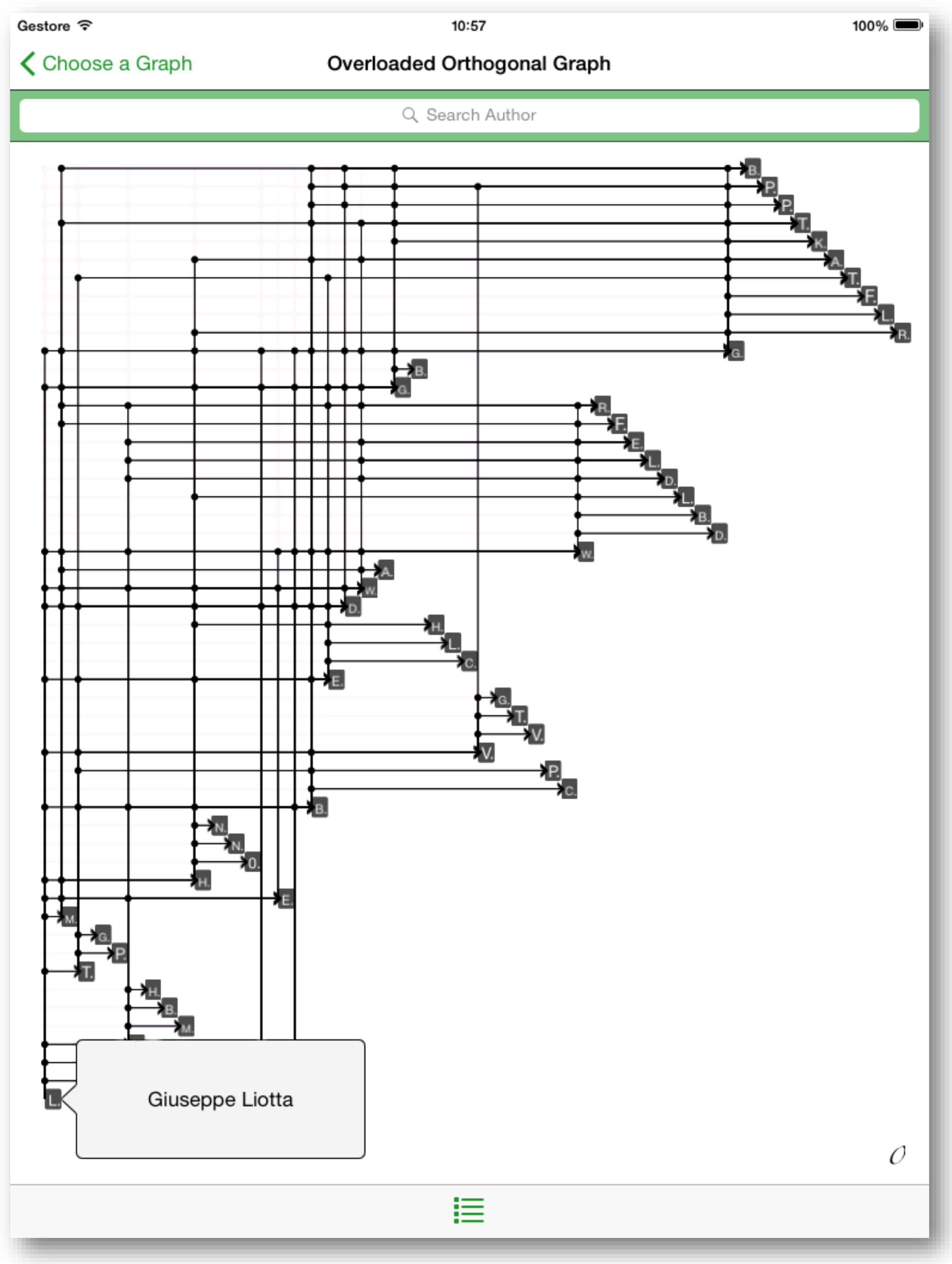
The system in 3 steps

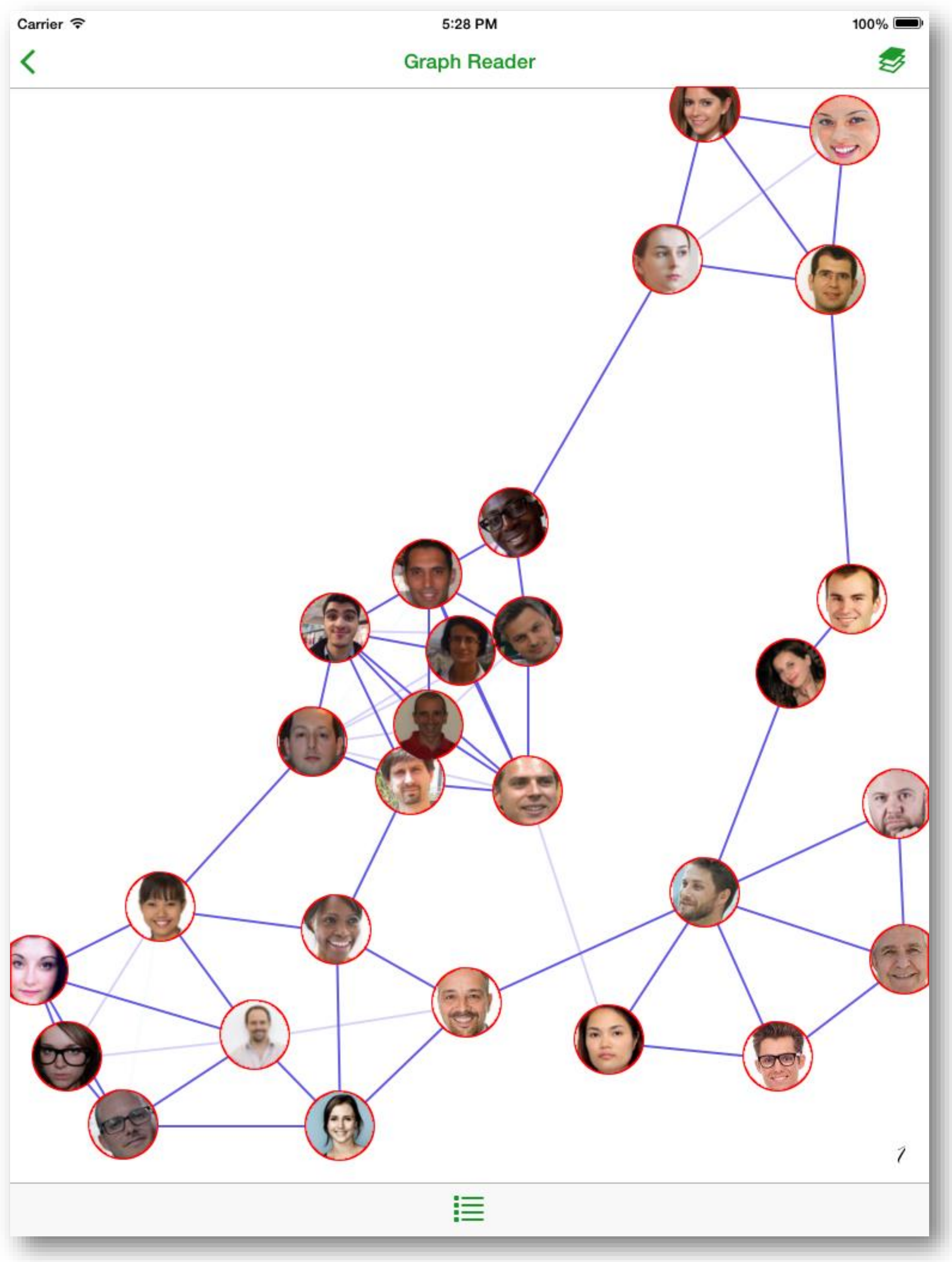
Sizing. This step takes as input Γ and returns a resized drawing Γ_R that matches the device display resolution and that guarantees all the drawing conventions of the input drawing Γ. Also, it should preserve as much as possible some of the drawing original prominent features. If the drawing is straight-line, a technique like the one described in [5] can be used. If the drawing is OOD, it is enough to properly adjust the horizontal and vertical grid unit of the drawing.

Paging. In this step we aim at partitioning the set of edges E into subsets $E_1, E_2, \dots, E_k$, such that the subdrawing $p_i = \Gamma_R[E_i]$ $(1 \leq i \leq k)$, called **page**, guarantees some desired property (in each subdrawing, the vertices remain fixed in their original positions as determined in Γ_R). The goal is to distribute the drawing visual complexity along the pages which can be explored separately. On the negative side, the user has to face the difficulty of making sense of a distributed information, thus it is important to have as few pages as possible. If Γ_R is straight-line, then a user could prefer to see each page p_i without any edge crossing, i.e., as planar, or that any two crossing edges form a sufficiently large angle. Computing a minimum set of pages that cover all the edges of G and such that each page guarantees a prescribed readability property can be heuristically computed in polynomial time [2]. If Γ_R is an OOD, a natural choice to build the set of pages is to assign to page p_i all the edges that represent shortest paths of length i.

Binding. Binding all the pages together is done by a set of interactive operations that can be used to browse the pages of the book and to explore a single page. To go from one page to the next one or to the previous one, it is enough to swipe the screen from left to right or from right to left, respectively: The edges in the current page will fade out, while the edges in the new page will fade in. Zooming in and out is done by the pinch and stretch gesture. The edges incident on a vertex, even those assigned to different pages, can be highlighted by tapping on the vertex. If tapping on two vertices the system shows the edge that connects the two vertices (if any). By selecting a region of the screen with a circle gesture the subdrawing induced by the vertices inside the region is highlighted.

A demo video is available at http://youtu.be/sMDmMkr3lfs





Two screenshots of the GraphBook system.

Top: The input is an OOD drawing of a DBLP coauthorship network and the displayed page shows the paths of length one in the network.

Bottom: The input is a straight-line drawing of a social network and the readability property used to compute the displayed page is large angle crossings.

1. G. Da Lozzo, G. Di Battista, and F. Ingrassia. Drawing graphs on a smartphone. JGAA, 16(1):109–126, 2012.
2. E. Di Giacomo, W. Didimo, G. Liotta, F. Montecchiani, and I. G. Tollis. Techniques for edge stratification of complex graph drawings. JVLC, 25(4):533–543, 2014.
3. P. A. Eades. A heuristic for graph drawing. In Congressus Numerantium, volume 42, pages 149–160, 1984.
4. E. M. Kornaropoulos and I. G. Tollis. Overloaded orthogonal drawings. In Graph Drawing, volume 7034 of LNCS, pages 242–253. Springer, 2011.
5. Y. Wu, X. Liu, S. Liu, and K.-L. Ma. Visizer: A visualization resizing framework. IEEE TVCG, 19(2):278–290, 2013.