**Motivation**

Because of its scale and its internet-mediated nature, the OMSCS program presents a number of unique challenges in the field of educational technology.

One of these is the practice of academic advising as it applies to the program. Dr. Charles Isbell, a Senior Associate Dean and one of the professors most heavily involved in the program thus far, has cited this as a key threat to the program’s scalability. It’s simply not feasible to support each and every student with a professional advisor (the way a more expensive “executive Master’s” program might) or a faculty advisor (the way a smaller research-oriented or PhD feeder program might) – nor will it ever be.

At present students perform “DIY advising” by posting questions about courses to the unofficial Google+ group and reading reviews shared in the unofficial course review survey. This is more scalable than e-mailing the advisor, but it requires a great investment of time on the part of both students and their more experienced peers.

With the recent departure of Mimi Haley, the program is back down to one academic advisor, with an advisor-advisee ratio of between 1:2000 and 1:3000 – the highest in the program’s history. It’s now more critical than ever to offer OMSCS students the planning information they need in an approachable, scalable and consistent format.

**Foundation**

[Advising as a discipline; recommendation engines as a discipline; existing tools for advising.]

There are two promising patterns we can apply here; both have yielded good results in other areas of the program. […]

**Vision**

*Advising*

The initial goal of the project is to provide advising services (specifically academic advising – career or other advising is outside the scope of the project), and to provide them in such a way that their quality improves rather than degrading as the program grows larger. This breaks down into four major components:[[1]](#footnote-1)

* User profile & history: allows the student to input (at a minimum) their previously-completed courses, desired specialization, timetable for graduation, and study hours per week. Eventually, may serve as the basis for social features (sharing one’s class schedule with others to plan study groups).
* Planner: considers the user’s profile and suggests appropriate courses and enrollment dates; allows the user to save their plan and tweak it later, or start over from scratch if they want to make a major change (e.g., taking a term off or switching specializations).
* Course browser: allows the user to access additional course information that may help in evaluating how appropriate the planner’s suggestions are for their unique interests and circumstances.
* Review system: allows the user to give feedback on courses they’ve completed, to benefit both the system (which will use some of this data, e.g. effort estimates, in recommendations) as well as other students (who will look at their predecessors’ opinions when deciding whether to take a course).

*Platform for research*

The OMSCS program is one of the largest course-based CS MS programs in the country, and the average class offers more than 200 seats per term. Moreover, the current group of OMSCS students has shown a high degree of “civic-mindedness”, a willingness to go to some lengths to enhance the program experience for others. These factors combine to present a unique opportunity for data collection.

I have already identified some features that are likely predictors for how much a student will like a course and/or how much value they will derive from it. In the long run, though, these give us merely a starting point; application of machine learning techniques could yield a far richer set of factors to consider for each student. For example, no official prerequisites are currently established for any course in the OMSCS program; does this make sense, or are there “unofficial” prerequisites one needs in order to do well in certain courses? Can we infer that if you do well in one course requiring Java, you’ll do well in another course requiring Java?

Conversely, can our data offer value to members of the OMSCS community beyond the student body? For instance, can we hope to advise professors on what factors make a course well-loved, or how many students are likely to enroll in their course for the upcoming term (and thus how many TAs they need)?

*Social features*

In recent years, advisors have begun to experiment with online communities of advisees. These communities are often built on course management platforms like Moodle and facilitate one-to-one or one-to-many communication. However, they rarely encourage interactions between peers. In addition, those rare communities that *do* encourage peer interactions are still largely formed around artifacts provided by the advisor (e.g., an advising syllabus) as opposed to material created by the students.

Because of the youth and size of the OMSCS program, its social landscape is still very much in flux. We’ve already outlined the reasons for pursuing a peer-driven overhaul of “traditional” advising functions; might the *advising community* benefit from a similar overhaul? Even very rudimentary social features, such as the ability to add “friends” and later see which friends are planning to take which classes when, might help students form smaller, more manageable study groups within larger classes.

Not only could these features benefit students directly; they would also enhance the advising and research functionality mentioned above. If a large number of students opted into the social elements, we could begin to approach such research questions as “How much does a student’s connectedness change their likelihood of dropping out?” and “Do students prefer taking classes with ‘high-profile’ peers?” We could also consider the presence of known study buddies as a factor influencing a student’s success in a course, and invite them to adjust their schedule accordingly.

*Future collaboration*

In the long term, the vision for omscs-advisor must evolve to more closely fit the needs of the OMSCS community. The clearest way to get there is by eventually open-sourcing the project or at least inviting code contributions from other students.

In order to attract and sustain a significant development community, I’ve made the decision to build the entire app on the Django web MVC framework. Django is a popular framework; it was one of the winners of my informal (and admittedly quite flawed) poll in the OMSCS Google+ group. In addition, because Django applications are written in Python, OMSCS students who participate in a Django project will be building useful skills that can be applied in many of the program’s courses.

This presents some challenges of its own, as I have never previously written a web application with Django! However, I believe my background in using other web frameworks (and recent work in Python) has prepared me to pick it up relatively quickly and with a minimum of fuss.

**Implementation**

*Scope*

All of the foregoing is nice, but because of the constrained duration of this course, it’s necessary to pick and choose somewhat. The absolute core functionality of the advisor is the ability to recommend courses, so that’s where we’ll start. (Where possible, I have tailored the UI design to allow for future extension in the other two areas.)

Furthermore, building an intelligent recommender system will take enough time that it cannot reasonably be shoehorned into this phase of the project. [algorithm details]

[statistics details]

The trimmed version of the project includes only 16 webpages. For a slightly more detailed picture of this planned interface, see the wireframes in *Appendix A*.

*Schedule*

The following is a proposed schedule for this phase of the omscs-advisor project.

|  |  |  |
| --- | --- | --- |
| Date | Theme | Deliverable(s) |
| 10/3 | Setup, ramp-up | Register omscs-advisor.com domain  Set up EC2 instance running LAMPython + Django  Complete Django tutorial  Create landing page, login page, and account creation page  Stuff code into omscs-advisor repository  Update license terms  Write ToC / privacy policy |
| 10/10 | Metadata | Encode metadata for specializations  Encode metadata for courses |
| 10/17 | Planner | Implement core planner UI  Allow user to set planner restrictions/preferences  Draft & send e-mail to key supporters inviting them to a pre-alpha test |
| 10/24 | Profile | Implement ability to save preferences  Implement ability to save plan |
| 10/31 | Profile, cont’d | Add ability to configure privacy settings  Add ability to view others’ profiles (if not private)  Write alpha user recruitment post & publish it to Piazza |
| 11/7 | Course browsing | Add course index (browse & search) page  Add course detail page  Film project video trailer |
| 11/14 | Course reviews | Add course review / “add to course history” interface  Account for already-completed courses in user plan  Add per-course statistics on course thumbnails and course details  Write beta user recruitment post & publish to Google+ groups |
| 11/21 | Course statistics | Finish course review importer (from UCRSS), populate legacy reviews  Adjust course recommendation algorithm to prioritize courses with higher rating |
| 11/28 | Review & retrospective | Final paper/presentation |
| 12/3\* | Preparation for future work | Solicit beta user feedback  Plan future research directions |

*Risk mitigation*

The project does not depend extensively on external integrations. While use of omscs-advisor is limited to students at Georgia Tech, integrating with IAP appeared too involved for the duration of this project – therefore, identity verification will be performed purely through e-mail address verification (i.e., students will be required to sign up with a Georgia Tech e-mail address). This avoids complex data use restrictions and scary FERPA requirements; it does require that the students manually enter any course history data they might wish the advisor to consider, but even this is not such a great burden since they should have fewer than 10 courses in total (far fewer than the ~50 for an undergraduate).

However, the project does suffer from another source of risk:

No project plan survives first contact with the user.

As the project is somewhat over-planned, this could hit it particularly hard. The plan calls for mitigating this risk through the use of staged rollouts:

* By 10/17, a hand-picked set of users will be solicited to give feedback on just the planner UI. (I identified these students during the Spring 2015 registration period, the Fall 2015 registration period, and the first few weeks of this class. They represent a small group very interested in the subject of course selection.)
* By 10/31, any users enrolled in Educational Technology will be solicited to try the initial version of the webapp.
* By 11/14, all users in the program will be invited to create accounts and give feedback.

In the event that beta users express dissatisfaction with the app, the work currently planned for the week of 11/21 and 11/28 can be swapped out for more in-demand features. No such accommodation is currently in the schedule to allow for major changes suggested by pre-alpha and alpha users.

**Appendix A: Wireframes**

(Note: unfortunately these are rather unprofessional-looking at the moment. For the final proposal draft they will be recreated in Balsamiq or Bootstrap.)

1. Note: I took this breakdown, plus the “motivation” section, from my own miniproposal. That in turn largely recycled the breakdown from my project readme at <https://github.com/Arkaaito/omscs-advisor>. [↑](#footnote-ref-1)