Fundamental & Computational Sciences

Advanced Comput, Math & Data Research Highlights

June 2014

Match. Color. Repeat.

Combinatorial graph techniques put schedulers in the driver's seat

When a frustrated car lover who just happens to have a keen interest in applied graph theory takes to Google, the search results can yield something interesting. In this case, one former professor's online search for an efficient way to judge classic Jaguars led him to Dr. Mahantesh Halappanavar, a scientist with Pacific Northwest National Laboratory's High Performance Computing group, whose research involving graph matching and coloring algorithms proved to be an opportune and novel matchup between classic cars and classical graph theory.

Judging a Thing of Beauty

As a Professor Emeritus in computer science and mechanical engineering at California State University, Fullerton and a current and longtime Jaguar aficionado, Dr. Edward F. Sowell faced a problem: when assigning judges to specific car classes as part of his duties as a Concours d'Elegance Judge for the Jaguar Owners Club, Los Angeles (affiliated with Jaguar Clubs of North America), he found the process could take up to 20 hours or more simply to work out the schedule.

Judging for the Concours events, which are hosted several times a year, has specific mandates. Every car belongs to a specific class under two main divisions, Driven and Championship, and each class requires a team of two or three judges, depending on the division. Because judges must undergo training, apprenticeship and certifications to be eligible to evaluate specific classes, there are only a limited number of available judges. Car owners, who also might be judges in other classes, must be present during the evaluation. All of these criteria ultimately result in a complicated scheduling problem that currently is resolved manually before each event.

A 1976 Jaguar XJ-S owned by Dr. Edward F. Sowell. *Photo courtesy of E. Sowell.* Enlarge Image.

Seeking the proverbial "better way," Sowell drew from his work on graph theory applications with Lawrence Berkeley National Laboratory's Simulation Research Group that developed the Simulation Problem Analysis and Research Kernel (funded by the U.S. Department of Energy) and started searching on Google, where he ended up connecting with PNNL's Halappanavar through his former Ph.D. advisor Professor Alex Pothen at Purdue University.

Through another DOE-funded project, Pothen and Halappanavar had worked on and developed software addressing the graph matching problem. However, after a few e-mail exchanges, Halappanavar realized Sowell's Jaguar judging dilemma posed a complex scheduling problem. One that forced him to look for a new

combinatorial solution—an approach that combined elements from his work in graph matching and graph coloring developed in the context of high-performance computing. Arif Khan, Pothen's graduate student, who at the time was an intern at PNNL, also assisted in the design and prototype implementation.

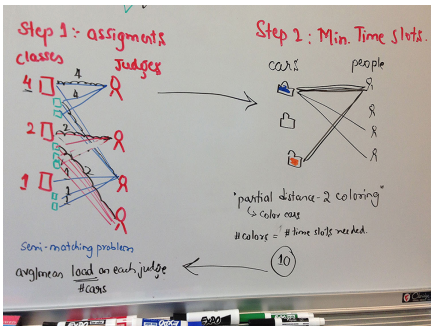
As a way of seeking balance for judges' schedules, the team formulated a two-step process: 1) assigning judges to car classes using weighted semi-matching and 2) assigning cars and people to a minimum number of time slots using partial distance-2 coloring. To the best of their knowledge, the team's pairing of these algorithms to compute an efficient schedule was the first effort of its kind.

"With the first step, we wanted to come up with an assignment that minimizes the average number of cars that a judge gets assigned. This is the semi-matching part of the problem," Halappanavar explained. "While in the second step, we compute the minimum number of time slots required to complete the day's judging based on the cars assigned per person. The total number of colors equals the number of time slots needed.

"The problem gets complicated because there are unequal numbers of cars in each class, and the same team needs to judge all the entries in a class," he continued. "Judges also can enter their own cars and as owners need to be present during judging of their cars. And, owners cannot be judges for their cars or any other cars in the classes they belong to. There are several other rules and restrictions that make it challenging."

Driven by Graph Theory

In this work, a graph, $G=(V, E)$, consists of a set of nodes, V, that represent entities and a set of edges, E. An edge represents a binary relationship between two nodes. The current algorithm works by first building a bipartite graph, where the node set can be partitioned into two parts, and the edges only can connect nodes in one set to nodes in the other.



Whiteboard depiction of the software at work, depicting the graph problem with its unique pairing of nodes—in this case, the pairing between judges and car classes—and coloring showing how any two cars sharing an owner or judge receive different colors. This was the framework used to test the program and build the schedule for the May 2014 Jaguar club's Concours event. Enlarge Image.

In the "Jaguar" test case, classes form the first set of nodes, and judges are the second. Each class then is replicated to accommodate the number of judges needed for that class based on the class's category. If a given judge is eligible to evaluate a particular class, an edge is added between the two nodes that represents the class and judge, respectively. A weight is assigned to the edge based on the number of cars entered in that class. A semi-matching computes an optimal solution that minimizes the mean number of cars assigned to judges.

Once provided an optimal assignment, a new bipartite graph is built with individual cars forming one set of the nodes and people (judges and owners combined) composing the other. Each car will have edges to its owner and two or three judges assigned to the car. Coloring is done in such a manner that any two cars sharing a person (owner or judge) receive different colors. This means that the two cars cannot be judged in the same time slot. Minimizing the number of colors used decreases the total number of time slots needed to complete the event.

By decoupling the process into two steps, the researchers were able to compute something very close to an optimal schedule in only a matter of minutes. For example, a slightly different optimal assignment from Step 1 might require slightly fewer colors. Now, the team is investigating techniques for fixing this problem, as well as other methods that can be used to compute efficient solutions.

"Our current work here at PNNL required semi-matching software for load-balancing tasks on HPC platforms for a molecular dynamics application, which helped us in solving the first step of the assignment problem," Halappanavar said. "Our partial distance 2-coloring work also benefits several machine learning projects under CASS [Center for Adaptive Supercomputing Software], such as stochastic coordinate descent and

community detection."

In the Field

After several conversations, tests and evaluations, an initial field trial of the software was rolled out in time for the Jaguar Owners Club of Los Angeles Annual Concours d'Elegance on May 18, 2014 in Fullerton, California. To prepare, Sowell sent Halappanavar data files with information regarding the event. The car information included data about owners, classes and categories, while information about judges included their eligibility and availability to evaluate different classes. Halappanavar fed the information into the program and let the algorithms go to work. Sowell used the resulting data, which Halappanavar delivered as a text file, to build a presentable schedule in Microsoft Excel that was used during the event. As part of the process, Halappanavar also made a few optimizations manually, and Sowell checked them for accuracy, editing them as necessary to meet other needs on the ground.

This immaculately restored, rare 1939 SS was such a hit that it caused the judging to run well beyond the allotted timeslot for the May 2014 Jaguar Concours event in Southern California. Adapting the software to accommodate these last-minute changes is another option being pursued. *Photo courtesy of E. Sowell*. Enlarge Image.

While on paper the event appeared fully workable, a few "variables" thrown into the mix during the actual event—judges suddenly unavailable and a late change to the schedule—showed the software requires a bit more work to make it fully viable on the fly.

"We need to provide some means of helping the Chief Judge make ‘safe changes' at the event," Sowell noted via e-mail. "Having done a lot of manual editing on the assignments prior to the event, I know it would not be difficult to develop an app that runs on a tablet to do things like ‘find a no-conflict time slot for this car/team that needs to be moved.' The app would work off of a text file from PNNL, downloaded to the tablet beforehand. The Chief Judge would then pencil the change onto his paper assignment sheet. I've started working on the idea."

Back in the Lab

Meanwhile, observations gleaned from the initial field test are helping Halappanavar to adapt the software and are generating a distinct test case that shows the software's potential practical applications.

"Our software only takes a fraction of a second to compute solutions, and we have already been able to solve some variants of matching and coloring problems on graphs with billions of edges in a matter of minutes to hours," he said. "The software scales well on several of PNNL's high-performance computing platforms."

In the future, Halappanavar and his colleagues expect to present their findings based on the Jaguar judging scenarios in a paper. For now, they are working on updates for another field test expected during the San Diego Jaguar Club's 50th Annual Concours d'Elegance in July.

"This complex scheduling challenge started last summer with a mathematically curious car owner," Halappanavar noted. "Now, it has an immediate benefit: it is a much faster way to build a schedule. There are many more problems that we expect to address using our graph algorithms and software."

Related Work

***On Matching*:**

Azad MA, M Halappanavar, S Rajamanickam, EG Boman, A Khan, and A Pothen. 2012. "Multithreaded Algorithms for Maximum Matching in Bipartite Graphs." In: *IEEE 26th International Parallel & Distributed Processing Symposium (IPDPS 2012)*, pp. 860-872. May 12-25, 2012, Shanghai, China. IEEE Computer Society, Los Alamitos, California. DOI: 10.1109/IPDPS.2012.82.

Halappanavar M, J Feo, O Villa, A Tumeo, and A Pothen. 2012. "Approximate weighted matching on emerging manycore and multithreaded architectures." *International Journal of High Performance Computing Applications* 26(4):413-430. DOI: 10.1177/1094342012452893.

Khan A, DF Gleich, A Pothen, and M Halappanavar. 2012. "A Multithreaded Algorithm for Network Alignment via Approximate Matching." In: *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, November 10-16, 2012, Salt Lake City, Utah. Institute of Electrical and Electronics Engineers, Piscataway, New Jersey. DOI: 10.1109/SC.2012.8.

***On Coloring*:**

Çatalyürek ÜV, J Feo, AH Gebremedhin, M Halappanavar, and A Pothen. 2012. "Graph coloring algorithms for multi-core and massively multithreaded architectures." *Parallel Computing* 38(10-11):576-594. DOI: 10.1016/j.parco.2012.07.001.

Scherrer C, M Halappanavar, A Tewari, and DJ Haglin. 2012. "Scaling Up Coordinate Descent Algorithms for Large l_1 Regularization Problems." In: *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, eds. J Langford and J. Pineau. June 26, 2012, Edinburgh, Scotland. International Machine Learning Society, Madison, Wisconsin.

Lu H, M Halappanavar, A Kalyanaraman, and S Choudhury. 2014. "Parallel heuristics for scalable community detection." In: *Proceedings of the International Workshop on Multithreaded Architectures and Applications (MTAAP)*, IPDPS Workshops, pp. 1375-1385. May 23, 2014, Phoenix, Arizona.