# Demonstrating Principled Uncertainty Modeling for Recommender Ecosystems with RecSim NG


M. Mladenov*
Google Research
Mountain View, CA
mmladenov@google.com

C. Hsu*
Google Research
Mountain View, CA
cwhsu@google.com

V. Jain
Google Research
Mountain View, CA
vihanjain@google.com

E. Ie
Google Research
Mountain View, CA
eugeneie@google.com

C. Colby
Google Research
Mountain View, CA
ccolby@google.com

N. Mayoraz
Google Research
Mountain View, CA
nmayoraz@google.com

H. Pham
Google Research
Mountain View, CA
huberpham@google.com

D. Tran
Google Research
Mountain View, CA
trandustin@google.com

I. Vendrov
Google Research
Mountain View, CA
ivendrov@google.com

C. Boutilier
Google Research
Mountain View, CA
cboutilier@google.com



## ABSTRACT

We develop RecSim NG, a probabilistic platform that supports natural, concise specification and learning of models for multi-agent recommender systems simulation. RecSim NG is a scalable, modular, differentiable simulator implemented in Edward2 and TensorFlow.



## CCS CONCEPTS

• **Computing methodologies → Simulation environments**; • **Information systems → Recommender systems**.

## KEYWORDS

Probabilistic Programming, Latent Variable Models, Reinforcement Learning

**ACM Reference Format:**
M. Mladenov, C. Hsu, V. Jain, E. Ie, C. Colby, N. Mayoraz, H. Pham, D. Tran, I. Vendrov, and C. Boutilier. 2020. Demonstrating Principled Uncertainty Modeling for Recommender Ecosystems with RecSim NG. In *Fourteenth ACM Conference on Recommender Systems (RecSys '20), September 21–26, 2020, Virtual Event, Brazil.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3383313.3411527


*Contact author.



*RecSys '20, September 21–26, 2020, Virtual Event, Brazil*



ACM ISBN 978-1-4503-7583-2/20/09.
https://doi.org/10.1145/3383313.3411527


## 1 INTRODUCTION

Recent years have seen increased emphasis, both in research and in practice, on recommender systems (RSs) that are capable of sophisticated interaction with users, beyond simply presenting items and passively observing immediate user reactions (clicks, consumption, ratings, purchase, etc.). This includes systems capable of exploring user interests [8, 16], optimizing engagement over multi-step horizons [7, 11, 13, 21, 27], or engaging in natural language dialogue [9, 23]. Such RSs cannot generally be trained using *static* data sets since assessing the effect of *counterfactual actions* on user behavior is crucial. Moreover, once we model non-myopic user behavior, the *interaction* between content consumers and content providers takes on added importance in predicting RS performance. Indeed, almost every practical RS embodies a *complex, multi-agent ecosystem.* This amplifies the need for RS methods to capture long-term behavior *of and among* participants, as well as RS models that capture these (potentially strategic) interactions.

To facilitate the development and study of RS methods in such complex environments, we develop RecSim NG, a configurable platform for both authoring and learning RS *simulation environments*. Effective simulation can be used to evaluate existing RS policies, or generate data to train new policies (in either a tightly coupled online fashion, or in batch mode). Just as simulation has greatly accelerated progress in reinforcement learning (RL) research [2, 4, 5], RecSim NG can support advances in long-horizon and ecosystem-aware RSs. Broadly, RecSim NG provides a probabilistic programming framework that allows the natural specification of user and creator behavior/dynamics within RS ecosystems, the ability to learn the parameters of such models from data, and supports a number of probabilistic inference techniques (beyond Monte Carlo (MC) simulation) for evaluating RS algorithms.

## 2 KEY GOALS AND CONTRIBUTIONS

We briefly outline the main aims and benefits of RecSim NG.

**User state dynamics.** Models of user (and provider) behavior over interaction *sequences* must reflect user *state* evolution. RecSim NG builds on the probabilistic programming language Edward2 [25] to specify common design patterns for user state and behavior (e.g., user preferences, satisfaction, choice/consumption behavior). RecSim NG emphasizes causal, generative models of user behavior and utility (e.g., user state, choice, & response/engagement models; user-state transition dynamics). These models are specified as a *composable* set of dynamic Bayesian networks (DBNs) [10, 15] organized in an object-oriented fashion [19] using three main concepts (or *EBSs*): *entities* (e.g., users, recommenders), *behaviors* (e.g., state transitions, choice models) and *stories* (e.g., user-system interaction details).

**Latent state models.** User models should reflect *latent* or unobservable user state. To support effective (long-term) recommendation, RSs must often engage in latent-state estimation, given various observable behaviors, to build internal models of (say): user preferences; user psychological state (e.g., satisfaction, frustration); and other exogenous environmental factors (e.g., user context such as activity). RecSim NG allows the flexible specification of observables so that practitioners can test the state estimation capabilities of different model architectures.

**General probabilistic inference.** Behavior models require flexibility in their structure and inference capabilities. For example, in psychometric models of user choice, structural priors and biases are often easy to specify, while precise parameterization is not. RecSim NG supports this flexibility, allowing one to impose model structure while learning model parameters (and structure if desired) from data. Because realistic models have latent factors, model learning requires sophisticated probabilistic inference, beyond the usual MC rollouts employed by other simulation environments. RecSim NG supports latent variable inference, including several MCMC and variational methods.

**Ecosystem modeling.** Capturing ecosystem effects is important to model long-term RS outcomes. This includes: incentives of agents that drive behavior; variable observability criteria for pairs/groups of agents; and the *interaction* between agents as mediated by the RS. The use of EBSs allows the natural specification of the interaction between agents in the system, while making it easy to uncover any independence that exists. Moreover, scalability is critical, especially in RSs with large populations. RecSim NG provides scalable TensorFlow-based execution to support several forms of posterior inference beyond standard MC rollouts.

We primarily view simulation as a tool to explore, evaluate, and compare different RS models, algorithms, and strategies. While the so-called "sim2real" perspective is valuable, we have largely used RecSim NG for simulations that reflect *particular* phenomena of interest to allow the controlled evaluation of RS methods at suitable levels of abstraction. RecSim NG should be equally valuable to researchers (e.g., as an aid to reproducibility and model sharing) and practitioners (e.g., to support rapid model refinement and evaluation prior to training in a live system).

RecSim NG is fully described in forthcoming white paper [18]. We demonstrate one simple use case below (for others, see, e.g., [3, 17]). RecSim NG is a significant extension of RecSim, an earlier platform with some of the same goals.[1] RecSim NG differs from RecSim in its flexibility, generality and ease of use (object-orientation, probabilistic programming); functionality (e.g., inference, ecosystem support); and scalability (TF-based execution, vectorized computation and GPU/TPU acceleration). RecSim NG will be released open-source prior to RecSys-2020 on GitHub and as a PyPI package, like RecSim. Researchers can download the code (and use tutorials) to create their own simulations.

RecSim NG bears some connection in form and motivation to other simulation platforms for RSs [20, 22, 26], RL [2, 5, 12, 24] and user ecosystem models [1, 28]. We elaborate on differences in the corresponding white paper [18],

## 3 DEMONSTRATION OF RECOMMENDER ECOSYSTEM DYNAMICS

We demonstrate the use of RecSim NG using a stylized ecosystem model that demonstrates the interaction between a large population of users and content providers as mediated by a content recommender system. The demo video can be found at https://bit.ly/30cb43P. A rough outline of the model is as follows (see details of a related model in [17]):

- Each content provider is represented by a "topicality" point in an embedding or *topic* space $S \subseteq \mathbb{R}^d$ reflecting the type of content offered by that provider. Each item lies in $S$. Providers generate random content at each round of engagement with the RS, drawn from a Gaussian centered at their topicality point. Provider state reflects the (past-discounted) cumulative user engagement with their items. This captures their incentive to participate in the RS ecosystem—lower engagement reduces (stochastically) the number of items offered at any round.
- Each user is characterized by two latent variables. A *interest vector* $u$ in topic space $S$ represents her general interests for content, and her reward for consuming an item $i$ is inversely proportional to $i$'s Euclidean distance to $u$. A satisfaction score reflects her past-discounted cumulative reward for items consumed. When a slate of items is recommended, the user selects an item using a multinomial logit choice model (w.r.t. item reward). User interest vectors are drawn at random using the following community model. Each user is associated with one content provider. The probability a provider is chosen for a given user is proportional to the distance of the provider's topicality point from the center of the topic space $S$. This emulates a niche vs. mainstream content divide, with topics closer to the center considered more mainstream. Once her community is sampled, the user's interest vector $u$ is sampled from a unit-variance Gaussian centered on that provider's topicality vector.
- At each round, each provider randomly offers content to the RS: providers with more (recency-weighted) user engagement have greater probability of generating more items. From the available content at each round, the RS uses some strategy (see below) to recommend a slate of $k$ items to a

[1] RecSim is available (open-source) at `github.com/google-research/recsim` and is detailed in [14].

user, from which the user selects using the choice model above. The RS cannot observe the latent state of any user (but it is is free to estimate it); but it can directly track each provider's engagement.

Our demonstration illustrates how to use RecSim NG to: (a) concisely and naturally specify the environment (users, providers, and their state and behaviors) using entities, behaviors and stories; (b) evaluate the performance of different RS policies using simulation; and (c) accomplish the latter effectively using distributed TensorFlow-based computation and the probabilistic independence identified by Edward2.

We evaluate two distinct RS policies. The first is a standard *myopic* policy used by traditional RSs: it matches a user to content that best matches her *estimated* interests. While this policy updates its estimate of a user's interest given her observed response, it does not explicitly *explore* to learn these interests. More critically, this policy in unaware of provider incentives—even if a provider is likely to reduce the amount of content it offers, the policy does not anticipate the effect this might have on future user satisfaction. In particular, providers of "niche" content—content that appeals to a small number of users—are likely to effectively drop out of the system, reducing the overall utility the RS provides to its audience by driving these users to more "mainstream" providers (from which niche user derive lower utility). Such "rich-get-richer" dynamics can explain various power-law phenomena in RSs [6].

The second RS strategy is a provider-aware policy that matches content to user "requests" in a way that explicitly accounts for the potential withdrawal of providers and *the impact this can have on long-term user satisfaction*. This policy matches users to a slightly more diverse set of providers, which keeps more providers engaged with the RS. The resulting increased viability of niche providers over long horizons generates greater population-level user satisfaction.

## REFERENCES

[1] J. Ahlgren, M. E. Berezin, K. Bojarczuk, E. Dulskyte, I. Dvortsova, J. George, N. Gucevska, M. Harman, R. LÃ¤mmel, E. Meijer, S. Sapora, and J. Spahr-Summers. 2020. WES: Agent-based User Interaction Simulation on Real Infrastructure. arXiv:2004.05363.

[2] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. 2013. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research* 47 (June 2013), 253–279.

[3] C. Boutilier, C.-w. Hsu, B. Kveton, M. Mladenov, C. Szepesvari, and M. Zaheer. 2020. Differentiable Bandit Exploration. (2020). arXiv:2002.06772.

[4] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. 2016. OpenAI Gym. (2016). arXiv:arXiv:1606.01540 preprint arXiv:1606.01540.

[5] P. S. Castro, S. Moitra, C. Gelada, S. Kumar, and M. G. Bellemare. 2018. Dopamine: A Research Framework for Deep Reinforcement Learning. (2018). arXiv:1812.06110.

[6] Ò. Celma. 2010. The Long Tail in Recommender Systems. In *Music Recommendation and Discovery*. Springer, 87–107.

[7] M. Chen, A. Beutel, P. Covington, S. Jain, F. Belletti, and E. Chi. 2018. Top-K Off-Policy Correction for a REINFORCE Recommender System. In *12th ACM International Conference on Web Search and Data Mining (WSDM-19)*. Melbourne, Australia, 456–464.

[8] K. Christakopoulou and A. Banerjee. 2018. Learning to Interact with Users: A Collaborative-Bandit Approach. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. San Diego, 612–620.

[9] K. Christakopoulou, F. Radlinski, and K. Hofmann. 2016. Towards Conversational Recommender Systems. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. ACM, New York, NY, USA, 815–824.

[10] T. Dean and K. Kanazawa. 1989. A Model for Reasoning about Persistence and Causation. *Computational Intelligence* 5, 3 (1989), 142–150.

[11] J. Gauci, E. Conti, Y. Liang, K. Virochsiri, Y. He, Z. Kaden, V. Narayanan, and X. Ye. 2018. Horizon: Facebook's Open Source Applied Reinforcement Learning Platform. (2018). arXiv:1811.00260 [cs.LG].

[12] S. Guadarrama, A. Korattikara, O. Ramirez, Pa. Castro, E. Holly, S. Fishman, K. Wang, E. Gonina, N. Wu, C. Harris, V. Vanhoucke, and E. Brevdo. 2018. TF-Agents: A Library for Reinforcement Learning in TensorFlow. https://github.com/tensorflow/agents. https://github.com/tensorflow/agents Online; accessed 25-June-2019.

[13] E. Ie, V. Jain, J. Wang, S. Narvekar, R. Agarwal, R. Wu, H.-T. Cheng, T. Chandra, and C. Boutilier. 2019. SlateQ: A Tractable Decomposition for Reinforcement Learning with Recommendation Sets. In *International Joint Conference on Artifical Intelligence (IJCAI)*. Macau, 2592–2599.

[14] E. Ie, C.-w. Hsu, M. Mladenov, V. Jain, S. Narvekar, J. Wang, R. Wu, and C. Boutilier. 2019. RecSim: A Configurable Simulation Platform for Recommender Systems. (2019). arXiv:1909.04847.

[15] U. Lerner, B. Moses, M. Scott, S. McIlraith, and D. Koller. 2002. Monitoring a Complex Physical System using a Hybrid Dynamic Bayes Net. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI-02)*. Edmonton, 301–310.

[16] S. Li, A. Karatzoglou, and C. Gentile. 2016. Collaborative Filtering Bandits. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. ACM, New York, NY, USA, 539–548. https://doi.org/10.1145/2911451.2911548

[17] M. Mladenov, E. Creager, K. Swerksy, O. Ben-Porat, R. S. Zemel, and C. Boutilier. 2020. Optimizing Long-term Social Welfare in Recommender Systems: A Constrained Matching Approach. In *Proceedings of the Thirty-seventh International Conference on Machine Learning (ICML-20)*. Virtual Conference. to appear.

[18] M. Mladenov, C.-w. Hsu, V. Jain, E. Ie, C. Colby, N. Mayoraz, H. Pham, D. Tran, I. Vendrov, and C. Boutilier. 2020. RecSim NG: Toward Principled Uncertainty Modeling for Recommender Ecosystems. (2020). in preparation.

[19] A. Pfeffer and D. Koller. 1997. Object-oriented Bayesian Networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-97)*. Providence, RI, 303–313.

[20] D. Rohde, S. Bonner, T. Dunlop, F. Vasile, and A. Karatzoglou. 2018. RecoGym: A Reinforcement Learning Environment for the problem of Product Recommendation in Online Advertising. (2018). arXiv:1808.00720 [cs.IR].

[21] G. Shani, D. Heckerman, and R. I. Brafman. 2005. An MDP-based Recommender System. *Journal of Machine Learning Research* 6 (2005), 1265–1295.

[22] J.-C. Shi, Y. Yu, Q. Da, Shi-Y. Chen, and A.-X. Zeng. 2019. Virtual-Taobao: Virtualizing Real-world Online Retail Environment for Reinforcement Learning. In *Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence (AAAI-19)*. Honolulu, 4902–4909.

[23] Y. Sun and Y. Zhang. 2018. Conversational Recommender System. (2018). arXiv:1806.03277 [cs.IR].

[24] Yuandong Tian, Q. Gong, W. Shang, Y. Wu, and C. L. Zitnick. 2017. ELF: An Extensive, Lightweight and Flexible Research Platform for Real-time Strategy Games. In *Advances in Neural Information Processing Systems 30 (NIPS-17)*. Long Beach, CA, 2659–2669.

[25] Dustin Tran, M. W. Hoffman, D. Moore, C. Suter, S. Vasudevan, and A. Radul. 2018. Simple, distributed, and accelerated probabilistic programming. In *Advances in Neural Information Processing Systems 31 (NeurIPS-18)*. Montreal, 7598–7609.

[26] X. Zhao, L. Xia, Z. Ding, D. Yin, and J. Tang. 2019. Toward Simulating Environments in Reinforcement Learning Based Recommendations. (2019). arXiv:1906.11462.

[27] X. Zhao, L. Xia, L. Zhang, Z. Ding, D. Yin, and J. Tang. 2018. Deep Reinforcement Learning for Page-wise Recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys-18)*. Vancouver, 95–103.

[28] Stephan Zheng, A. Trott, S. Srinivasa, N. Naik, M. Gruesbeck, D. C. Parkes, and R. Socher. 2020. The AI Economist: Improving Equality and Productivity with AI-Driven Tax Policies. arXiv:2004.13332.