**1. Intro**

* Heterogeneity has become a fundamental property of the spectrum
* Coexistence techniques are narrow, complex, and short-lived
* Heterogeneous spectrum management can be an efficient long-term solution
* Organizing heterogeneous spectrum is non-trivial with such high density
  + Need to overlap technologies intelligently
* Current models and spectrum assignment algorithms mainly homogeneous
  + Unable to capture heterogeneous properties or impact of heterogeneous networks operating together
  + Need to predict/estimate technologies overlapping
* We present a novel framework for heterogeneous spectrum management
  + Metric to predict performance of heterogeneous networks together
  + Optimization to re-organize the spectrum towards greater efficiency
  + Optimization takes in to consideration reconfigurability of devices

**2. Requirements in Heterogeneous Spectrum Assignment**

What is needed + a hint of why related work falls short

**2.1.** **An Appropriate Model**

* Need a model that is able to represent heterogeneous environment and various different properties across PHY/MAC
* Graph is still OK 🡪 but needs to be more informative (e.g., uni-directional edges) and more heavily annotated

**2.2. Speculative/Predictive Performance Metric**

* Spectrum is too dense to simply “separate” technologies. Avoiding interference completely is likely impossible, so you want to instead place things to minimize it.
* Problem is NP-hard, can’t expect devices to keep trying various configurations. Need to predict their performance in different parts of the spectrum together.
* Prior work (e.g., WifiNet) has made such predictive metrics for heterogeneous impact on Wifi only. Metric needs to be more comprehensive across all technologies (i.e., generic)

**2.3. Spectrum Assignment Algorithm**

* Using the model *and* predictive performance metric, try various configurations which can have different objective functions (e.g., maximize spectrum utilization, minimize worst-case heterogeneous interference, etc.)
* Spectrum assignment algorithm *needs to take in consideration possible configurations.* I.e., needs to consider which devices can be reconfigured and not, their possible operational frequencies, and their possible sets of bandwidths (e.g., 20MHz channels vs. 40MHz, etc). Much of this comes from the more highly annotated model which gives us the necessary information.

***Should related work go here? Can’t decide, might distract flow to our model…***

**3. Heterogeneous-Aware Spectrum Assignment**

Here we present our model, algorithm etc. Following the structure of section 2, we present our model, techniques, and algorithm to perform heterogeneous spectrum management.

**3.1. An Appropriate Heterogeneous Graph**

* From the underlying monitoring system (e.g., WifiNet, Airshark, anything…), our system takes information and constructs the following type of graph which is a hypergraph
* Graph uses uni-directional links to *importantly* model interference/conflict and asymmetry.
* On each of the edges in the graph is a 0 or 1 which the underlying monitoring system (and/or other tools) annotate to specify whether the behavior between the two interfaces is cooperative (sharing) or conflicting (destructive interference)
* Finally, the “hyper” aspect of the graph is a special type of edge (i.e., a hyperedge) which connects devices in the network which have a *network dependency* i.e., they must be reconfigured in the same way. This means, they all must share the same chosen frequency (e.g., all devices on the same 802.11 network)… or sets of frequencies if dualband.
* Along with this basic information in the graph, each node (device) is annotated with its possible set of frequencies, possible sets of bandwidths, its reconfigurability. This is later used by the algorithm to know how to configure the devices.

**3.2.**  **Heterogeneous-Aware Channel Evaluation (HCE)**

* The metric we propose is generic and uses observed loads of networks and their cooperation (or not) to predict their performance on channels
* HCE accounts for two things: 1) Expected fair share of airtime with nodes you would cooperate with, and 2) Your expected degradation or loss in airtime due to sustained heterogeneous interference.
* We calculate sustained heterogeneous interference using the following underlying assumption: nodes that do not cooperate transmit entirely independently. They do not backoff. I.e., they can be modeled as poisson processes and using their expected airtimes, we can predict (roughly) how often they would have conflicting transmissions. This is generic enough for *any* technology.
* <Insert the mathematical HCE derivation of sustained interference>
* **Need a microbenchmark to show that we can roughly predict how often they overlap…** doesn’t need to account for how often the overlaps are destructive (i.e., packet loss) 🡪 that is more complex based on modulations, etc. Our assignment algorithm and metric allows such information to be considered (i.e., not every overlap is a loss) if the monitoring system provides it to be more accurate.

**3.3. MLP Formalization and Optimization of Spectrum Management**

* We present our MLP formalization of the heterogeneous spectrum management problem.
* The formalization uses the hypergraph/model as input to the configuration possibilities, general input
* The formalization uses HCE to continually predict the outcome of the placing various networks and devices in different parts of the spectrum, accounting for coordination and expected loss
* <Insert the formalization details>

**4. Simulation of Model and Optimization**

* Use various different styles of environments. For example, one in the home which represents heterogeneous and a lot of unconfigurable results (e.g., apartment complex)
* A home scenario which is more isolated where mostly everything is reconfigurable
* A enterprise like environment where most reconfigurable devices are the Wifi devices (btw—WifiNet NEVER actually did reconfiguration of their Wifi network with their estimation of interference, so this is novel despite WifiNet).
* Now, use different objective functions from the optimization, show how can increase/decrease fairness and/or efficiency.

**5. Real World Evaluation**

* Hopefully take 3 small environments and show that the algorithm produces reasonable results which improves performance

**6. Related Work**

**7. Conclusions**