Work on any analysis that works on top of the control flow graph, such as:

Uniqueness analysis: when things are used only once. Say you map, then you filter, then you map again, you create a lot of intermediaries

Independence analysis:

Usage analysis:

Take things that work on a list and make it work destructively.

Shape analysis.

Paper by Jurian Haage on reusing data structures

A generic usage analysis with subeffect qualifiers

<https://www.researchgate.net/publication/221241231_A_generic_usage_analysis_with_subeffect_qualifiers>

Model some usage analysis as a contract in Kotlin.

Keywords: Usage analysis, reusing data structures, uniqueness analysis, uniqueness typing,

You need to start somewhere with saying how often something is used, say in for example a map function

<https://link.springer.com/chapter/10.1007/3-540-46425-5_11>

using types for resource aware programming

<https://dl.acm.org/doi/10.1145/3453483.3454032>

garbage collection algorithm that encourages reuse

Think of a few example programs where these things would work.

Casting a mutable list to immutable and back

Report is usually about 15 pages, 3 pages to explain problem, 10 pages intro to Kotlin, features that need to be known for the language, a few pages with a rough outline with experiments idea, motivating examples, maybe prototype, a planning (not too detailed). Specific problems that need to be tackled for the general case, literature survey