Repair techniques.

* Search space size problem
  + Large ->intractable
  + Small -> desired repair may not in the search space
* Correctness guarantees.
* Promising avenue is leveraging other development artifacts through open source software.

Complexed programs and patches

Search-based program techniques moved from reasoning over populations of programs AST to populations of small edit programs(the patches themselves)

Constraint-based have moved from reasoning about the semantics of entire methods to only reasoning about the desired change in behavior

->These efforts enable scaling to programs of significant size, and multi-line repairs.

1. APR 본질적인 기술에 대한 평가를 하기위해 FL은 모두 완벽한 위치를 제공
2. 시간에 대한 평가를 할 때 preparation은 모두 제외하고 FL이후 repair과정에서의 소모시간을 측정(preparation은 한 번 셋팅 이후부터는 반복적으로 소모되는 값이 아니기 때문
3. Category : Search-based, pattern-based, (constraint-based?) ,learning-based,

* Jgenprog, Cardumen, Tbar, Confix, sequencer, coconut.
* Jgenprog : statement level search-based algorithm(ingredient:Package)
* Cardumen : expression level search-based algorithm(ingredient Global)
* Sequencer : Learning-based nmt model based rnn
* Coconut : Learning-based nmt model based cnn
* Tbar : pattern-based algorithm
* Confix : context-based algorithm

미래에 일반적으로 쓰일 기반 기술은 pattern based, learning-based techniques.

실험 test set : coconut과 Tbar가 배치되는 test set 구성해서 비교분석, pattern-based 는 희소하게 발생되는 버그타입에 관해 fix 확률이 learning based에 비해 ?

Pattern-based : 새 패턴에 대응하려면 추가적인 manually work 필요

Learning-based : 학습의 랜덤 적인 부분

Jgenprog,cardumen은 기존 파일에 대해 ingredient를 추출해서 기반으로 fix하기 때문에 코드상에 없는 형태의 수정 형태면 고치기 힘듬.

Confix 와 Tbar모두 template-based기반하지만 Tbar는 조금 더 직접적인 pattern에 대입하고 confix는 수집된 모든 경우의 context를 체크하기 때문에 quality 문제 보유