Term	Explanation
SMT (Satisfiability Modulo Theories) Solver	Checks validity or satisfiability of logical formulas automatically. Example solvers are Yices from SRI and Z3 from Microsoft.
Hypothesis Space	Defines the search space of candidate functions that the synthesizer has to search over. Can be infinite.
Oracle	A human or an automated solver which answers questions posed by the synthesizer.
Interaction model	The way the synthesizer and the oracle interact. Concretely, refers to particular kinds of questions asked by the synthesizer, and answers given by the oracle.
CEGIS (Counter-Example Guided Inductive Synthesis)	Refers to a particular interaction model between a synthesizer and an oracle, where the oracle provides counter-examples (so the oracle is more than just a classifier which says yes/no, but also gives a reason for the failure).
Domain Specific Language (DSL)	A programming language targeting particular application domain. Typically used to specify the hypothesis space
Initial Specification	The initial specification provided to the synthesizer. This may be a logical spec or just a set of examples.

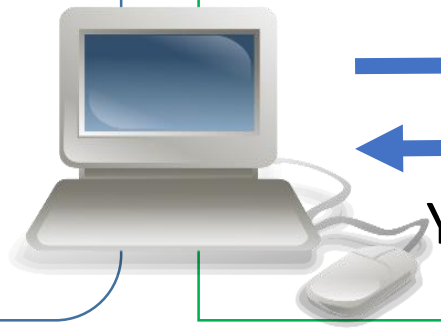
	Covered in	Initial Specification	Hypothesis Space	Interaction Model	Supported Oracle Queries	Who is the Oracle?	Is SMT used?
Sketch	Lecture 2, Lecture 3, slide 17	Naïve implementation	Programs with "holes"	CEGIS	Equivalence – oracle returns 'yes', or counter-example	SAT/SMT solver	yes
Flash-Fill	Lecture 3, slides 7-13	A set of input-output examples	DSL of string manipulation functions	None	None	None	no
Oracle-guided	Lecture 3, slide 17	None	Programs which are permutation of a predefined set of components	Synthesizer poses membership queries to oracle	Membership – oracle returns yes/no	Human	Yes, to find: P, P', in
SPEX	Lecture 3, slides 18-44 Exercise 3	None/one input-output example	Disjunctive/Conjunctive Formulas	Synthesizer poses membership queries to oracle	Membership – oracle returns yes/no	Human	No

Sketch + CEGIS

Oracle is often implemented by a constraint-solver, e.g., SMT

Sketch

```
bit[W] isolate0Sketched(bit[W] x){
  return ~(x + ??) &  (x  + ??);
}
```



CEGIS

```
bit[W] isolate0Fast (bit[W] x)
  return ~x & (x+1);
```

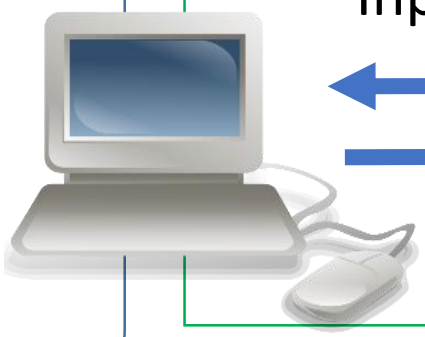
Yes/counterexample

Flash Fill

User provides a set of examples already, so it does not use an oracle to answer queries. It also does not use an SMT solver.

PBE Synthesizer

$$
\begin{array}{rcl}
\text{String expr } P & := & \texttt{Switch}((\mathrm{b}_1, \mathrm{e}_1), \cdot\cdot, (\mathrm{b}_n, \mathrm{e}_n)) \\
\text{Bool b} & := & \mathrm{d}_1 \vee \cdot\cdot \vee \mathrm{d}_n \\
\text{Conjunct d} & := & \pi_1 \wedge \cdot\cdot \wedge \pi_n \\
\text{Predicate } \pi & := & \texttt{Match}(v_i, \mathrm{r}, k) \mid \neg\, \texttt{Match}(v_i, \mathrm{r}, k) \\
\text{Trace expr e} & := & \texttt{Concatenate}(\mathrm{f}_1, \cdot\cdot, \mathrm{f}_n) \\
\text{Atomic expr f} & := & \texttt{SubStr}(v_i, \mathrm{p}_1, \mathrm{p}_2) \\
 & \mid & \texttt{ConstStr}(s) \\
 & \mid & \texttt{Loop}(\lambda w : \mathrm{e}) \\
\text{Position p} & := & \texttt{CPos}(k) \mid \texttt{Pos}(\mathrm{r}_1, \mathrm{r}_2, \mathrm{c}) \\
\text{Integer expr c} & := & k \mid k_1 w + k_2 \\
\text{Regular Expression r} & := & \texttt{TokenSeq}(\mathrm{T}_1, \cdot\cdot, \mathrm{T}_m) \\
\text{Token T} & := & C+ \mid [\neg C]+ \\
 & \mid & \text{SpecialToken}
\end{array}
$$

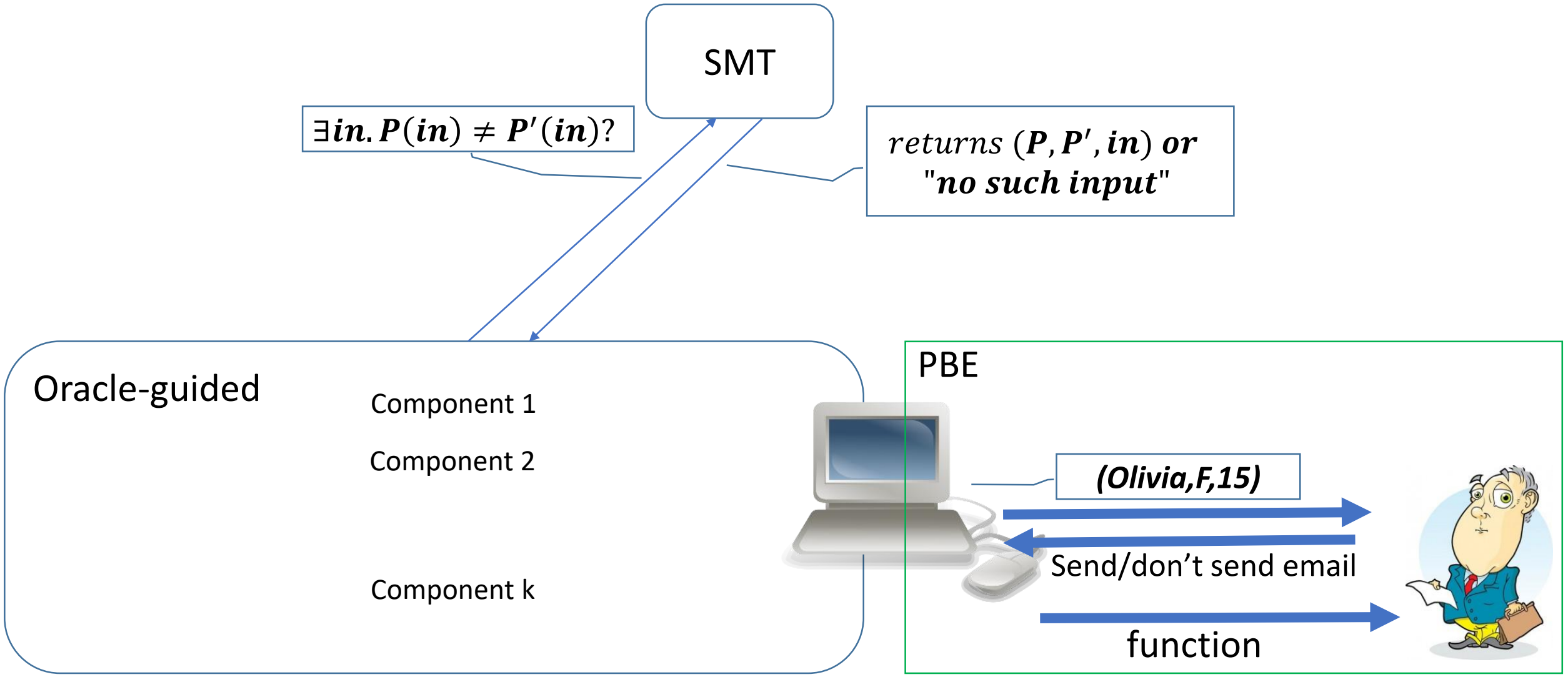


PBE

Input-output examples

function

Oracle-guided (Jha et al., 2010)



Exact PBE

Hypothesis space and interaction model are such that one can provide guarantees on the number of questions.

Exact PBE Synthesizer

A set of predicates Q +

$$
\begin{array}{rcl}
\text{String expr } P & := & \texttt{Switch}((\mathrm{b}_1, \mathrm{e}_1), \cdot\cdot, (\mathrm{b}_n, \mathrm{e}_n)) \\
\text{Bool b} & := & \mathrm{d}_1 \vee \cdot\cdot \vee \mathrm{d}_n \\
\text{Conjunct d} & := & \pi_1 \wedge \cdot\cdot \wedge \pi_n \\
\text{Predicate } \pi & := & \texttt{Match}(v_i, \mathrm{r}, k) \mid \neg\, \texttt{Match}(v_i, \mathrm{r}, k) \\
\text{Trace expr e} & := & \texttt{Concatenate}(\mathrm{f}_1, \cdot\cdot, \mathrm{f}_n) \\
\text{Atomic expr f} & := & \texttt{SubStr}(v_i, \mathrm{p}_1, \mathrm{p}_2) \\
 & \mid & \texttt{ConstStr}(s) \\
 & \mid & \texttt{Loop}(\lambda w : \mathrm{e}) \\
\text{Position p} & := & \texttt{CPos}(k) \mid \texttt{Pos}(\mathrm{r}_1, \mathrm{r}_2, \mathrm{c}) \\
\text{Integer expr c} & := & k \mid k_1 w + k_2 \\
\text{Regular Expression r} & := & \texttt{TokenSeq}(\mathrm{T}_1, \cdot\cdot, \mathrm{T}_m) \\
\text{Token T} & := & C+ \mid [\neg C]+ \\
 & \mid & \text{SpecialToken}
\end{array}
$$

Exact PBE