UNIVERSITAT POLITÈCNICA DE VALÈNCIA

VICERRECTORADO DE INVESTIGACIÓN, INNOVACIÓN Y TRANSFERENCIA

IMPRESO 1: MEMORIA CIENTIFICO-TÉCNICA Y PRESUPUESTO

INVESTIGADOR RESPONSABLE DEL PROYECTO:

Sergio Jiménez Celorrio

TÍTULO DEL PROYECTO:

Artificial Intelligence for the Automated Synthesis and Validation of Programs

ÁREA TEMÁTICA EN LA QUE SE INSCRIBE:

Ciencias de la Computación y Tecnología Informática

RESUMEN *(Debe ser breve y preciso, exponiendo sólo los aspectos más relevantes y los objetivos propuestos):*

Programming is nowadays a manual handcraft task however, the need to automate programming tasks increases every day: Programming errors, commonly known as *bugs*, cause undesired software behavior making programs crash or enabling malicious users to access private data. Just for 2017, the cumulative cost of software bugs is worldwide estimated in more than one trillion US dollars.

This research project investigates a novel approach for software development, that leverages *Artificial Intelligence* (AI), to increase the automation of the programming process and hence, reduce the chances of introducing software bugs. The project proposes to address **automatic program synthesis and validation starting from** *input-output* **tests cases, and using** *AI planning* **as a problem solving engine**. Our current work on this particular research topic is the recipient of the *2016 distinguished paper award* at IJCAI (the main international conference on Artificial Intelligence).

VLC/CAMPUS
VALENCIA, INTERNATIONAL
CAMPUS OF EXCELLENCE

EMAS
Gestión
ambiental
verificada
REG.NO. ES-CV-000208

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

**VICERRECTORADO DE INVESTIGACIÓN,
INNOVACIÓN Y TRANSFERENCIA**

MEMORIA CIENTIFICO-TÉCNICA

---

1. **Historial científico-técnico del Equipo de Investigadores (Apartado 5.1,bases de la convocatoria)**
   - *Se valorará el historial científico del IP y del resto del equipo de investigación, valorando positivamente la participación de investigadores en formación y de algún investigador senior.*
   - *Los miembros del Equipo Investigador deberá ser de un mínimo de 3 miembros, incluido el solicitante*
   .

Máximo **dos páginas**

The project will be developed within the *Group of Reasoning on Planning and Scheduling* (GRPS, http://users.dsic.upv.es/grupos/grps/), that is part of the *Department of Computer Systems and Computation* of the UPV (DSIC, www.dsic.upv.es).

In the last 10 years, the GRPS has led three research projects of the Spanish national research plan on *AI planning*. Further, the GRPS has developed its own AI planner, that is participating at the International Planning Competition (IPC-2018, http://ipc2018.bitbucket.io).

The research group for developing this project would comprise three researchers: 1 principal investigator, 1 senior researcher and 1 PhD student. All of them are GRPS members.

1.1 Principal Investigator: Sergio Jiménez Celorrio

Sergio is a *Ramón y Cajal* fellow at the Universitat Politècnica de València (2017-2022). Previously, Sergio was a post-doc at the University of Melbourne and a *Juan de la Cierva* fellow at the Universitat Pompeu Fabra de Barcelona. From 2004 to 2013 Sergio was assistant lecturer at the Universidad Carlos III de Madrid where he obtained the *Distinguished Thesis Award 2011*. Sergio is co-organizer of the 7th *International Planning Competition* and program committee at top international conferences on Artificial Intelligence (including ICAPS that is the main international conference on *AI planning*).

His research addresses complex *decision-making* problems with innovative integrations of two *Artificial Intelligence* paradigms: *Machine Learning* and *AI Planning*. His research trajectory on this topic has produced a novel approach to the synthesis of programs, grammars and controllers for autonomous behavior and is the recipient of the *2016 distinguished paper award* at the International Joint Conference on Artificial Intelligence (IJCAI)[1], the main international conference on *Artificial intelligence*.

His top five scientific contributions are:

1. Inductive programming with AI planning [1], [2], [3].

2. Generalization in AI planning [4], [5], [6].

3. Learning to relieve the *knowledge acquisition bottleneck* in AI planning [7], [8], [9], [10], [11].

4. Machine learning for addressing the *curse of dimensionality* in AI planning [12], [13], [14].

5. Evaluation in AI planning [15], [16], [17].

General quality indicators of scientific research: **382 citations** (296 since 2013) and **h-index 11**. *Source google scholar.*

---

[1] IJCAI is ranked as *A+* by the CORE rank and with *h5-index* 43 at Google Scholar

1.2 Senior Researcher: Eva Onaindia

Eva Onaindia is a *Full Professor* at the Universitat Politècnica of València. She leads the GRPS where she conducts research in Artificial Intelligence, Automated Planning, Causal Model Learning and Recommender Systems.

Eva has published numerous articles in top-tier specialized conferences such as the International Joint Conference on Artificial Intelligence (IJCAI), Autonomous Agents and Multi Agent Systems (AAMAS), International Conference on Planning and Scheduling (ICAPS) or European Conference on Artificial Intelligence (ECAI), among others; and scientific journals related to topics of Artificial Intelligence, Planning, Argumentation and Recommender Systems such as IEEE Intelligent Systems, Information Sciences, Engineering Applications of Artificial Intelligence, Knowledge and Information Systems or Applied Intelligence, among others.

She has led several projects from the *National research plan* as well as served on multiple ICAPS, AAAI, IJCAI and ECAI conferences as a member of the Program Committee and as a senior member. Eva is Editor-in-Chief of the journal AI Communications, a Review Board member of the journal Applied Intelligence and she will be Program co-chair of ICAPS 2019 (the main international conference on AI planning).

Eva Onaindía has supervised so far a total of nine PhD theses and she is currently supervising four doctoral students. She has also participated in many diverse scientific evaluation committees at both national and international level. In the national scene, she has collaborated with National agencies ANEP and ANECA and she has been member of different expert boards and scientific committees. At an international level, she has participated in the evaluation of research projects for Israel, Italy or Czech Republic.

Eva has also participated in several R&D and Innovation contracts: with the firm DESTINIA S.L. (https://destinia.com/) for the development of a tourist recommender system, and currently in the construction of a system for *electricity price forecasting* aimed at energy savings (Ayuntamiento Lliria) using neural networks and data analysis techniques.

General quality indicators of scientific research: **1536 citations** (992 since 2013) and **h-index 22**. *Source google scholar.*

1.3 Researcher in training period: Diego Aineto

Diego Aineto is a PhD student at the *Department of Computer Systems and Computation* of the Universitat Politècnica of València.

Since 2016 Diego is holding a four-year FPU grant on *Causal discovery for predictive analysis* (FPU16/03184). Diego is also involved on the current research project TIN2017-88476-C2-1-R, funded by the Spanish national research plan and with title, *Reconocimiento de actividades y planificación automática para el diseño de asistentes inteligentes.*

With Eva Onaindia and Sergio Jiménez, Diego is already co-author of one paper on the International Conference on Planning and Scheduling (ICAPS) [7], and one journal paper [18]. Both papers on research topics related to the development of this project.

3

Vicerrectorado de Investigación, Innovación y Transferencia
Universitat Politècnica de València
Edificio 3A. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 03, ext. 71030 • Fax +34 96 387 79 29, ext. 77929
viit@upv.es
www.upv.es/viit

Primeros Proyectos de Investigación (PAID-06-18)

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

VICERRECTORADO DE INVESTIGACIÓN,
INNOVACIÓN Y TRANSFERENCIA

2. Estado del arte y objetivos específicos (Apartado 5.2.i, bases de la convocatoria)
- *Deben tratarse aquí: los antecedentes y estado actual de los conocimientos científicos-técnicos relacionados con el proyecto, incluyendo la bibliografía más relevante.*
- *Se valorará el conocimiento del estado actual del tema y la definición clara y viable de los objetivos específicos.*

*Máximo **tres páginas** incluyendo referencias.*

This research project will **investigate the integration of *AI planning* into the *Test Driven Development* paradigm for the automatic synthesis and validation of programs**. Here we briefly introduce this technology:

- **AI Planning (AIP)** is the Artificial Intelligence component that studies the synthesis of sets of actions to achieve some given objectives [19]. AIP arose in the late '50s from converging studies into *combinatorial search*, *theorem proving* and *control theory* and now, is a well formalized paradigm for problem solving with algorithms that scale-up reasonably well. State-of-the-art planners are able to synthesize plans with hundreds of actions in seconds time [20]. The mainstream approach for AIP is *heuristic search* with heuristics derived automatically from the problem representation [21, 22]. Current planners add other ideas to this like *novelty exploration* [23], *helpful actions* [24], *landmarks* [25], and *multiqueue best-first search* [26] for combining different heuristics.

- **Test driven development (TDD)** [27] is a popular paradigm for software development that is frequently used in *agile methodologies* [28]. In TDD, test cases are created before the program code is written and they are run against the code during the development, e.g. after a code change via an automated process. When all tests pass, the program code is considered *complete* while when a test fails, it pinpoints a *bug* that must be fixed from the program code. Tests cases are a natural form of program specification, programmers often claim *'code that is difficult to test is poorly written'*. Further, tests alert programmers of bugs before handing the code off to clients (the cost of finding a bug when the code is first written is considerably lower than the cost of detecting and fixing it later).

Program synthesis is the task of computing a program that satisfies a given formal specification. In 2008 the PhD Thesis work by Armando Solar-Lezama, at the University of California Berkeley, showed that it is possible to encode program synthesis problems in Boolean logic and therefore, use *satisfiability modulo theories* [29] to automatically compute programs [30]. Since then, there has been a surge of practical interest in the idea of program synthesis in the formal verification community and related fields [31]. To illustrate this, in 2013, a unified framework for program synthesis problems was proposed and since 2014 there is a yearly program synthesis competition, comparing the different algorithms for program synthesis in a competitive event (http://www.sygus.org). Further the US National Science Foundation is funding the ExCAPE research project[2], to transform the way programmers develop software by advancing the theory and practice of software synthesis (since 2012 the ExCAPE research project is funded with a total amount of $3,750,000$).

Now we review the two most successful approaches for automated program synthesis.

- **Programming by Example (PbE)**. PbE techniques have already been deployed in the real world and are part of the FLASH FILL feature of Excel

---
[2] https://www.nsf.gov/awardsearch/showAward?AWD_ID=1138996

in Office 2013 that generates programs for string transformation [32]. In this case the set of synthesized programs are represented succinctly in a restricted Domain-Specific Language (DSL) using a data-structure called version space algebras [33]. The programs are computed with a domain-specific search that implements a divide and conquer approach.

- In **Programming by Sketching (PbS)** programmers provide a partially specified program, i.e. a program with the high-level structure of an implementation but that leaves low level details undefined to be determined by the synthesizer [34]. This form of program synthesis relies on a programming language called SKETCH, for sketching partial programs. PbS implements a counterexample-driven iteration over a synthesize-validate loop built from two communicating SAT solvers, the inductive synthesizer and the validator, to automatically generate test inputs and ensure that the program satisfy them. Despite, in the worst case, program synthesis is harder than NP-complete, this counterexample-driven search terminates on many real problems after solving only a few SAT instances [35].

On the other hand, *program validation* is the task of proving, or disproving, the correctness of a given program with respect to a certain formal specification or property. Program validation is considered a necessary step for program synthesis. *Model checking* is the mainstream AI approach for the formal validation of programs and controllers [36]. In model checking a given model of a system (the program or controller to validate), is exhaustively and automatically checked to verify whether this model meets a given specification. Typically the specification contains safety requirements such as the absence of deadlocks and similar critical states that can cause the system to crash. Both the model of the system and the specification are formulated in a formal language. Current approaches for model checking reduces to graph search. Instead of enumerating reachable states one at a time, the state space can sometimes be traversed more efficiently by considering large numbers of states at a single step. For instance, representing set of states and transition relations as logical formulas or binary decision diagrams, like in *symbolic model-checking* [37].

Closely related to the aims of the project is the work on the automated synthesis of *Finite State Controllers* (FSCs) [38]. The state-of-the-art algorithms for computing FSCs follow a *top-down* approach that interleaves *programming* the FSC with validating it [39]. To keep the computation of FSCs tractable, the space of possible solutions is bound by the maximum size of the FSCs. The computation of FSCs includes works that compile this task into another forms of problem solving so they benefit from the last advances on off-the-shelf solvers (e.g. *classical planning* [40], *conformant planning* [41], *CSP* [42] or a *Prolog program* [43]). The validation and synthesis of programs from examples are research questions also addressed in the classic AI field of *Inductive Logic Programming* (ILP) [44, 45]. ILP arises from the intersection of *Machine Learning* and *Logic Programming* and deals with the development of inductive techniques to learn logic programs from examples and background knowledge, that are expressed as logic facts.

(Please find the full list of *references* at the end of the document)

3. Metodología y Plan de Trabajo (Apartado 5.2.ii, bases de la convocatoria)

- *Se deben detallar y justificar con precisión la Metodología y el Plan de Trabajo, incluyendo el desglose de tareas para alcanzar los objetivos propuestos. En cada tarea se debe especificar el tiempo previsto y las personas que deben realizarla, si es posible con la inclusión de un cronograma.*
- *Si solicita personal debe justificar su necesidad y las tareas que va a desarrollar.*
- *Se valorará la adecuación del plan de trabajo y la correcta distribución de tareas entre los distintos miembros del equipo.*

*Máximo **cinco páginas***

Our current research already shows that AI planners can synthesize programs for non trivial tasks like sorting lists, traversing graphs or manipulating strings [3, 40, 46, 39]. Table 1 reports the time invested by the AI planner FD [25] to solve the following programming tasks: computing the nth term of the *summatory* and *Fibonacci* series, *reversing* a list, *finding* an element (and the *minimum* element) in a list, *sorting* a list and traversing a binary tree.

Programming Task	Time (seconds)
Summatory	1
Fibonacci	5
Reverse	22
Find	336
Minimum	284
Sorting	30
Tree	165

Tab. 1: Time to synthesize the programs with the AI planner FD [25] on a processor *Intel Core i5 3.10GHz x 4* and with a 4GB memory bound.

3.1 Synthesis and validation of TDD programs as AI planning

Given a *TDD programming* task, our approach is modeling and solving it as it were an *AI planning* task.

Briefly the **state variables** of this AIP task are *fluents* of the kind:

- `var:=value`, that encode the program variables.
- `program(line):=instruction`, that encode the instructions at the different program lines.
- `pcounter:=line`, encoding the current program line.

The **initial/goal states** of the AIP task encode the initial/final values of the program variables and are given by the input/output tests of the TDD programming task.

Finally, program instructions are encoded using two kinds of AIP **actions**:

- *Programming actions*, that assign an instruction to a given program line.
- *Execution actions*, execute the instruction assigned to the current program line.

We implement this encoding using standard planning languages, such as PDDL [47], so the AIP tasks resulting from our encoding can be solved with off-the-shelf planners, like the FD planning system [25]. The programs synthesized with this approach are guaranteed to be bug-free over given sets of *input-output* tests cases.

Interestingly, our PDDL encoding allows also program validation by (1), specifying the lines of the program to validate (i.e. the `program(line):=instruction` fluents) in the initial state of the AIP task and (2), disabling the mentioned *programming actions* so only *execution actions* are applicable.

3.2 Evaluation

The performance of our approach for the synthesis and validation of TDD programs will be evaluated in two kinds of programming tasks:

- *Real-world benchmarks*: The GRPS group is currently leading the four-year research project TIN2017-88476-C2-1-R from the *Spanish national plan* in which *AI Planning* and *activity recognition* is applied to different real-world domains such as *domotics*, *tourism*, *traffic control* and *robotics*. Interestingly many activities in these domains can be understood as simple programs. For instance, Figure 1 shows a four-line program (pictured as a finite state machine) that represents the sequence of instructions required for the *making a buttered toast* activity. We plan to evaluate our approach in synthesis and validation tasks coming from these real-world domains.
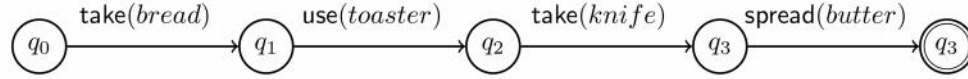

Fig. 1: Four-line program representing the activity of *making a buttered toast*.

- *Theoretical benchmarks*: Classic programming tasks are a neat touchstone to assess the performance of our approach. For instance, programs for the computation of mathematical/logic series, string manipulation and for the management of data structures such as *lists*, *queues*, *stacks* or *trees*. Figure 2 shows a synthesized program for traversing a linked list. The program is pictured as a *finite state machine*: The machine nodes mount to the different program lines while edges are tagged with a *condition/instruction* label, that denotes the condition (over the program variables) under which program instructions are taken.
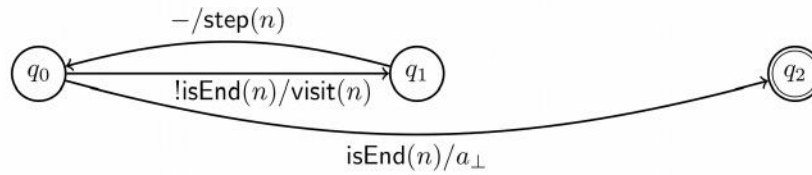

Fig. 2: Synthesized program to solve the programming task of traversing a linked list (n is a variable that points to a node of the linked list).

3.3 Workplan

We designed a 24-month workplan plan for the **development of a user-interactive program synthesizer** that (1), takes as input a set of test cases that specify the TDD programming task to solve and (2), outputs a program source code that passes these tests with a bug-free guarantee.

In the particular case that the *program synthesizer* receives an additional input specifying the program source code, it outputs a validation certificate

guaranteeing that the input program passes the given test cases. Figure 3 details the proposed 24-month timeline for the project. A deliverable is provided at the end of each task.
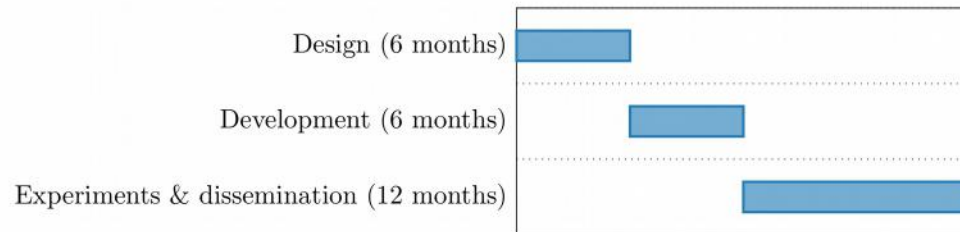

Fig. 3: Work-plan for developing a user-interactive program synthesizer/validator.

1. **T1. System design (months 1-6)**

   (a) Design of the test-case specification. Programming tasks are specified as a set of *input-output* tests cases plus the available instruction set.

   (b) Experimental design. Experiments will comprise taking time and memory measurements to evaluate the resources required by our approach to solve the given TDD programming/validation tasks.

   (c) Evaluation of the different AI planners available, with special attention to the planners that get the best results at the IPC-2018.

   Deliverable T1: Technical report with the specifications of the system design.

2. **T2. Development of the system architecture (months 7-12)**

   (a) The programming-into-planning compiler (*Compiler 1*). This system component parses the *TDD programming task* and produces an *AIP task* encoded in the standard planning language PDDL.

   (b) The plan-into-program compiler (*Compiler 2*). This system component extracts the program code and the corresponding validation certificate from the solution plan produced by an off-the-shelf AI planner.
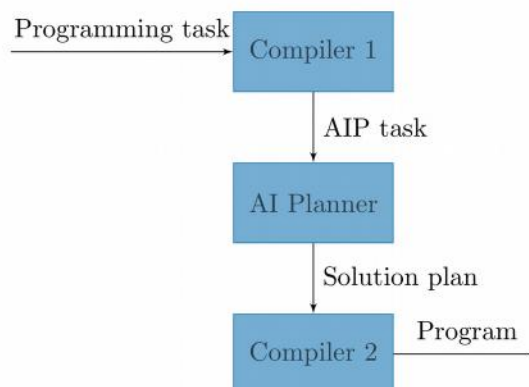

Fig. 4: System architecture for the synthesis and validation of TDD programs.

8

Deliverable T2: Open repository with the source code of the system architecture and the corresponding benchmarks.

3. **T3. Experiments and dissemination of results (months 13-24).**

   (a) Reporting the experimental performance of our AIP approach for solving diverse TDD programming tasks. This task will follow an iterative workflow over the following subtasks:

       i. Executing the system architecture in the *theoretical benchmarks* described in Section 3.2.
       ii. Analysis and validation of the obtained results.
       iii. Tuning and repairing the system components, evaluation metrics and benchmarks according to the obtained results.
       iv. Executing the system architecture in the *real-world benchmarks* introduced in Section 3.2.
       v. Analysis and validation of the obtained results.
       vi. Tuning and repairing the system components, evaluation metrics and benchmarks according to the obtained results.

   (b) Dissemination of the obtained theoretical and empirical results by submitting papers to top international conferences and journals in AI.

   Deliverable T3: Final report with the obtained conclusions and produced publications.

3.4 Workload

*Tasks (***T1-T3***)* will be developed by the three mentioned members of the research group: Sergio Jiménez, Eva Onaindia and Diego Aineto.

In addition, we plan to hire a master student for 6 months to assist in the completion of two tasks, **T3(a,i)** and **T3(a,iv)**. The detailed responsability of the student will be:

1. Executing the scripts of the system architecture.

2. Collecting the output data.

3. Presenting the output data in a easy-readable format.

VICERRECTORADO DE INVESTIGACIÓN, INNOVACIÓN Y TRANSFERENCIA

---

4. Objetivos específicos del Proyecto (Apartado 5.2.iii, bases de la convocatoria)

- *Se valorará la claridad de ideas en el desarrollo del proyecto y la novedad y relevancia de los objetivos en relación con el estado del conocimiento propio del área en el que se inscribe la solicitud.*

*Máximo **dos páginas***

The specific objective of this project is to study the performance of our AIP approach for the automated synthesis and validation of programs that are characterized by these tree dimensions:

1. Number of *program lines*.

2. Number and domain of the observable *program variables*.

3. Size of the available *instruction set*.

For instance the program of Figure 2, for traversing linked lists, is generated using three program lines, one observable variable (isEnd(n) that has Boolean domain and indicates when n points to the end of the given linked list), and an instruction set that comprises two instructions (visit(n), that marks the list node assigned to n as *visited* and step(n), that advances n to the next node in the linked list).

In more detail, the two specific objectives of this project are:

1. To study the performance of our Artificial Intelligence approach for the **synthesis of programs** up to: **10 program lines**, **10 observable variables** with binary domain and, an instruction set that comprises **10 instructions**.

2. To study the performance of our Artificial Intelligence approach for the **validation of programs** up to: **15 program lines**, **15 observable variables** with binary domain and, an instruction set that comprises **15 instructions**.

In both cases the performance of our approach will be evaluated with regard to the **computation time** and **memory** invested in the synthesize and the validation of the aimed programs.

Despite setting bounds for the programs size and kind, challenging programming tasks can be addressed using *problem decomposition*. With this regard, our AIP encoding already supports callable procedures to decompose a given programming task into simpler modules and to enable recursive solutions [40, 39].

5.   Beneficios del proyecto. Difusión y Explotación en su caso de los Resultados. (Apartado 5.3, bases de la convocatoria)

- ▪ *Se valorará la adecuación del Plan de difusión y explotación respecto de las contribuciones científico-técnicas esperables del proyecto.*
- ▪ *Impacto y contribuciones científico-técnicas esperables del proyecto. Se valorará positivamente el alineamiento con los ODS.*
- ▪ *Si pudiera derivarse impacto ambiental de consideración, se incluirá en la memoria el correspondiente estudio de impacto ambiental.*

*Máximo **una página***

AIP has recently shown successful in *program testing* to generate *attack plans* that completed non-trivial software security tests [48, 49, 50, 51]. Promising research opportunities come from the application of AIP to *program synthesis* given that, *program synthesis* with a tests base, can be seen as the *program testing* dual.

In fact, our current work on *program synthesis* with AIP already produced several **publications at top international conferences on Artificial Intelligence** [52, 46, 39, 40] and is the recipient of the *2016 distinguished paper award* at the International Joint Conference on Artificial Intelligence, the main international conference on *Artificial intelligence*.

The main benefit of this project is to provide new insights into the current understanding of how AI can assist programmers in the software development. In more detail, the expected benefits for this particular research project are four-fold:

1. A new **evaluation methodology for the *Synthesis and Validation of Programs***. The application of the exiting AIP technology to program synthesis can provide new evaluation metrics that assess how well a program covers a set of *input-output* test cases.

2. An empirical **study on the performance of the state-of-the-art AI planners for the *Synthesis and Validation of Programs***. Research in AI algorithms is too often tested with laboratory problems and AIP is not an exception. Most of the new planning algorithms are only tested within the benchmarks of the International Planing Competition [53]. This project will help to meet the computational and expressiveness limits of AI planners when addressing real-world programming tasks.

3. The **development of open software and open benchmarks for the *Synthesis and Validation of Programs***. We strongly belief that reproducibility and open knowledge are essential to the advance of the research on computer science. With this regard, we plan to develop a *github* repository where we make available the developed source code and benchmarks.

4. **International dissemination of the obtained scientific results**. The scientific results obtained during the development of the project will be submitted to top Artificial Intelligence conferences (such as IJCAI, AAAI, ICML and ICAPS) and to the main journals in the AI field (such as AIJ, JMLR and JAIR).

VLC/CAMPUS
VALENCIA, INTERNATIONAL CAMPUS OF EXCELLENCE

EMAS
Gestión ambiental verificada
REG.NO.ES-CV-000200

References

[1] J. Segovia-Aguas, S. Jiménez Celorrio, and A. Jonsson, "Generating context-free grammars using classical planning," in *International Joint Conference on Artificial Intelligence*, 2017.

[2] J. Segovia-Aguas, S. Jiménez Celorrio, and A. Jonsson, "Generalized planning with procedural domain control knowledge," in *International Conference on Automated Planning and Scheduling*, 2016.

[3] S. Jiménez Celorrio and A. Jonsson, "Computing plans with control flow and procedures using a classical planner," in *Eighth Annual Symposium on Combinatorial Search*, 2015.

[4] J. Segovia-Aguas, S. Jiménez Celorrio, and A. Jonsson, "Unsupervised classification of planning instances," in *International Conference on Automated Planning and Scheduling*, 2017.

[5] D. Lotinac, J. Segovia, S. Jiménez Celorrio, and A. Jonsson, "Automatic generation of high-level state features for generalized planning," in *International Joint Conference on Artificial Intelligence*, 2016.

[6] J. Segovia, S. Jiménez Celorrio, and A. Jonsson, "Hierarchical finite state controllers for generalized planning," in *International Joint Conference on Artificial Intelligence*, 2016.

[7] D. Aineto, S. Jiménez Celorrio, and E. Onaindia, "Learning strips action models with classical planning," in *International Conference on Automated Planning and Scheduling*, 2018.

[8] S. Jiménez Celorrio, F. Fernández, and D. Borrajo, "Integrating planning, execution and learning to improve plan execution," *Computational Intelligence*, vol. 29, no. 1, pp. 1–36, 2013.

[9] S. Jiménez Celorrio, F. Fernández, and D. Borrajo, "The pela architecture: integrating planning and learning to improve execution," *AAAI Conference on Artificial Intelligence*, 2008.

[10] S. Jiménez Celorrio, A. Coles, and A. Smith, "Planning in probabilistic domains using a deterministic numeric planner," *25th Workshop of the UK Planning and Scheduling Special Interest Group*, 2006.

[11] J. Lanchas, S. Jiménez Celorrio, F. Fernández, and D. Borrajo, "Learning action durations from executions," *Workshop on Planning and Learning ICAPS 2007*, 2007.

[12] S. Jiménez Celorrio, T. De La Rosa, S. Fernández, F. Fernández, and D. Borrajo, "A review of machine learning for automated planning," *The Knowledge Engineering Review*, vol. 27, no. 04, pp. 433–467, 2012.

[13] T. De la Rosa, S. Jiménez Celorrio, R. Fuenteteja, and D. Borrajo, "Scaling up heuristic planning with relational decision trees," *Journal of Artificial Intelligence Research*, pp. 767–813, 2011.

[14] T. De La Rosa, S. Jiménez Celorrio, and D. Borrajo, "Learning relational decision trees for guiding heuristic planning," *International Conference on Automated Planning and Scheduling*, 2008.

[15] C. L. López, S. Jiménez Celorrio, and Á. G. Olaya, "The deterministic part of the seventh international planning competition," *Artificial Intelligence*, vol. 223, pp. 82–119, 2015.

[16] C. L. López, S. Jiménez Celorrio, and M. Helmert, "Automating the evaluation of planning systems," *AI Communications*, vol. 26, no. 4, pp. 331–354, 2013.

[17] A. Coles, A. Coles, A. G. Olaya, S. Jiménez Celorrio, C. L. López, S. Sanner, and S. Yoon, "A survey of the seventh international planning competition," *AI Magazine*, vol. 33, no. 1, pp. 83–88, 2012.

[18] E. Onaindia, D. Aineto, and S. Jiménez, "A common framework for learning causality," *Progress in Artificial Intelligence*, pp. 1–7, 2018.

[19] M. Ghallab, D. Nau, and P. Traverso, *Automated Planning: theory and practice*. Elsevier, 2004.

[20] H. Geffner and B. Bonet, "A concise introduction to models and methods for automated planning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 8, no. 1, pp. 1–141, 2013.

[21] D. V. McDermott, "A heuristic estimator for means-ends analysis in planning.," in *International Conference on Artificial Intelligence Planning and Scheduling*, vol. 96, pp. 142–149, 1996.

[22] B. Bonet and H. Geffner, "Planning as heuristic search," *Artificial Intelligence*, vol. 129, no. 1, pp. 5–33, 2001.

[23] G. Frances, M. Ramırez, N. Lipovetzky, and H. Geffner, "Purely declarative action representations are overrated: Classical planning with simulators," in *IJCAI*, 2017.

[24] J. Hoffmann, "Ff: The fast-forward planning system," *AI magazine*, vol. 22, no. 3, p. 57, 2001.

[25] M. Helmert, "The fast downward planning system.," *Journal of Artificial Intelligence Research*, vol. 26, pp. 191–246, 2006.

[26] S. Richter and M. Westphal, "The lama planner: Guiding cost-based anytime planning with landmarks," *Journal of Artificial Intelligence Research*, vol. 39, no. 1, pp. 127–177, 2010.

[27] K. Beck, *Test-driven development: by example*. Addison-Wesley Professional, 2003.

[28] D. Cohen, M. Lindvall, and P. Costa, "Agile software development," *DACS SOAR Report*, no. 11, 2003.

[29] C. W. Barrett, R. Sebastiani, S. A. Seshia, C. Tinelli, *et al.*, "Satisfiability modulo theories.," *Handbook of satisfiability*, vol. 185, pp. 825–885, 2009.

[30] A. S. Lezama, *Program synthesis by sketching*. PhD thesis, Citeseer, 2008.

[31] R. Alur, R. Bodik, G. Juniwal, M. M. Martin, M. Raghothaman, S. A. Seshia, R. Singh, A. Solar-Lezama, E. Torlak, and A. Udupa, "Syntax-guided synthesis," in *Formal Methods in Computer-Aided Design (FMCAD), 2013*, pp. 1–8, IEEE, 2013.

[32] S. Gulwani, "Automating string processing in spreadsheets using input-output examples," in *ACM SIGPLAN Notices*, vol. 46, pp. 317–330, ACM, 2011.

[33] T. M. Mitchell, "Generalization as search," *Artificial intelligence*, vol. 18, pp. 203–226, 1982.

[34] A. Solar-Lezama, L. Tancau, R. Bodik, S. Seshia, and V. Saraswat, "Combinatorial sketching for finite programs," *ACM SIGOPS Operating Systems Review*, vol. 40, pp. 404–415, 2006.

[35] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[36] E. M. Clarke, O. Grumberg, and D. Peled, *Model checking*. MIT press, 1999.

[37] K. L. McMillan, "Symbolic model checking," in *Symbolic Model Checking*, pp. 25–60, Springer, 1993.

[38] B. Bonet and H. Geffner, "Policies that generalize: Solving many planning problems with the same policy," *IJCAI*, 2015.

[39] J. Segovia-Aguas, S. Jiménez, and A. Jonsson, "Hierarchical finite state controllers for generalized planning," in *International Joint Conference on Artificial Intelligence*, pp. 3235–3241, 2016.

[40] J. Segovia, S. Jiménez, and A. Jonsson, "Generalized planning with procedural domain control knowledge," in *International Conference on Automated Planning and Scheduling*, pp. 285–293, 2016.

[41] B. Bonet, H. Palacios, and H. Geffner, "Automatic derivation of finite-state machines for behavior control," in *AAAI Conference on Artificial Intelligence*, 2010.

[42] C. Pralet, G. Verfaillie, M. Lemaître, and G. Infantes, "Constraint-based controller synthesis in non-deterministic and partially observable domains.," in *ECAI*, 2010.

[43] Y. Hu and G. De Giacomo, "A generic technique for synthesizing bounded finite-state controllers," in *International Conference on Automated Planning and Scheduling*, 2013.

[44] S. Muggleton, "Inductive logic programming," *New generation computing*, vol. 8, no. 4, pp. 295–318, 1991.

[45] L. De Raedt, *Logical and relational learning*. Springer Science & Business Media, 2008.

[46] D. Lotinac, J. Segovia-Aguas, S. Jiménez, and A. Jonsson, "Automatic generation of high-level state features for generalized planning," in *International Joint Conference on Artificial Intelligence*, pp. 3199–3205, 2016.

[47] M. Fox and D. Long, "Pddl2. 1: An extension to pddl for expressing temporal planning domains.," *Journal of Artificial Intelligence Research*, vol. 20, pp. 61–124, 2003.

[48] J. Hoffmann, "Simulated penetration testing: From "dijkstra" to "turing test++"," in *Twenty-Fifth International Conference on Automated Planning and Scheduling*, 2015.

[49] M. Steinmetz, J. Hoffmann, and O. Buffet, "Revisiting goal probability analysis in probabilistic planning.," in *International Conference on Automated Planning and Scheduling*, pp. 299–307, 2016.

[50] D. Shmaryahu, "Constructing plan trees for simulated penetration testing," in *The 26th International Conference on Automated Planning and Scheduling*, vol. 121, 2016.

[51] M. Steinmetz, J. Hoffmann, and O. Buffet, "Goal probability analysis in probabilistic planning: Exploring and enhancing the state of the art," *Journal of Artificial Intelligence Research*, vol. 57, pp. 229–271, 2016.

[52] J. Segovia-Aguas, S. Jiménez, and A. Jonsson, "Generating context-free grammars using classical planning," in *International Joint Conference on Artificial Intelligence*, 2017.

[53] M. Vallati, L. Chrpa, M. Grześ, T. L. McCluskey, M. Roberts, S. Sanner, *et al.*, "The 2014 international planning competition: Progress and trends," *AI Magazine*, vol. 36, no. 3, pp. 90–98, 2015.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

VICERRECTORADO DE INVESTIGACIÓN, INNOVACIÓN Y TRANSFERENCIA

PRESUPUESTO
• El presupuesto deberá estar referidos al periodo de duración de la ayuda. Se deberá priorizar las necesidades económicas del grupo investigador, señalando los conceptos de gasto del 1 al 5

PERIODO DE DURACION DEL PROYECTO: (a elegir una opción)
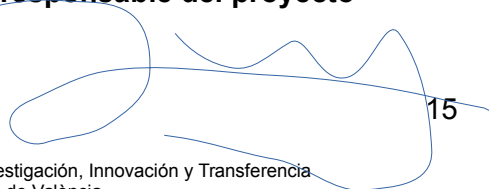
☐12 meses   ☐18 meses   X **24 meses**

ORDEN PRIORIDAD	CONCEPTO GASTO Y Justificación de la Necesidad	Importe
	GASTOS DE FUNCIONAMIENTO	2000
1	**Gastos adquisición material fungible: (máximo 20 %)**	-
	Gastos material inventariable: (máximo 20 %) Justificación: *High-performance Desktop Computer to run and evaluate the performance of the developed system.*	2000
	GASTOS DE PERSONAL:	5997
3	**Personal de Apoyo (máximo 60 %):** Justificación: *6-month, partial time (20 hours/week) contract for master student to assist in the completion of subtasks T3(a,i) and T3(a,iv)* • *Retribución mensual: 629,00 €* • *Importe total de la contratación: 5.996,93 €*	
	GASTOS DE VIAJES:	2000
2	**Colaboración con otros investigadores relacionada con el proyecto o asistencia a conferencias. (máximo 40 %)** Justificación: *Attendance to ICAPS, the main international conference on AI Planning.*	
	GASTOS DE DIFUSIÓN (Publicación en abierto)	-
	Gastos de difusión (máximo 60%) Justificación: Top international AI conferences such as AAAI, IJCAI, ICAPS and top journals like JMLR and JAIR are open access	
	TOTAL:	9997

Valencia a 15 de Junio de 2018

Investigador responsable del proyecto
Firma:

15

Vicerrectorado de Investigación, Innovación y Transferencia
Universitat Politècnica de València
Edificio 3A. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 03, ext. 71030 • Fax +34 96 387 79 29, ext. 77929
viit@upv.es
www.upv.es/viit

VLC/ CAMPUS
VALENCIA, INTERNATIONAL
CAMPUS OF EXCELLENCE

EMAS
Gestión ambiental verificada
REG.NO ES-CV-000091