دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

تفسیر انتزاعی برای تحلیل ایستای امنیت جریان اطلاعات

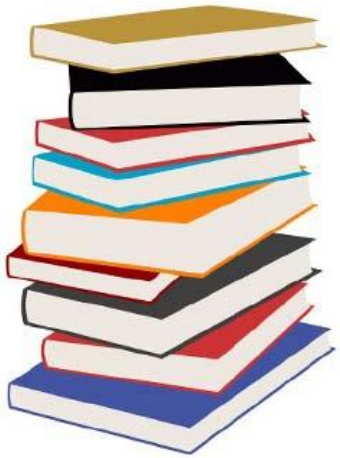
ارائه شفاهی آزمون جامع مقطع دکتری

سید محمدمهدی احمدپناه
smahmadpanah@aut.ac.ir

استاد راهنما: دکتر مهران سلیمان‌فلاح

دانشگاه صنعتی امیرکبیر
۱۲ دی ۱۳۹۷

دانشکده مهندسی کامپیوتر
و فناوری اطلاعات

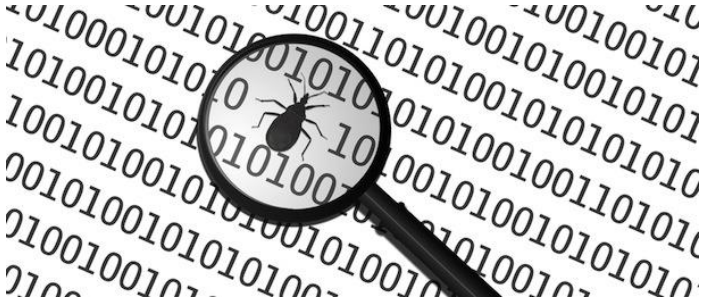


فهرست

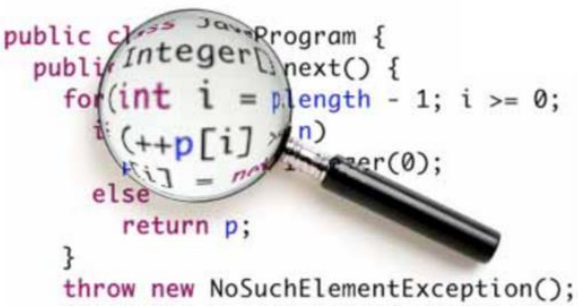
- مروری بر تحلیل برنامه
- معرفی تفسیر انتزاعی
- خط‌مشی‌های امنیتی و روش‌های اعمال آن‌ها
- به‌کارگیری تفسیر انتزاعی در اعمال ویژگی‌ها و فراویژگی‌ها
- چالش‌ها و مسائل باز
- جمع‌بندی
- لیست کنفرانس‌ها و مجله‌های مرتبط

مقدمه

- انواع روش‌های درستی‌سنجی نرم‌افزار [۱]
  - تحلیل پویا و آزمون
  - راستی‌آزمایی صوری
- انواع روش‌های راستی‌آزمایی صوری [۲]
  - روش‌های استنتاجی
  - وارسی مدل
  - تحلیل ایستا



تحلیل برنامه



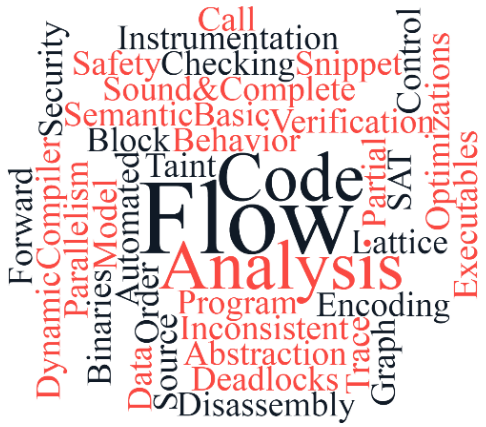
- تعریف تحلیل برنامه [۳]
  - تکنیک‌های خودکار زمان-کامپایل برای پیش‌بینی تقریب‌های ایمن و قابل محاسبه از مجموعه مقادیر یا رفتار زمان‌اجرای یک برنامه
- کاربردهای تحلیل برنامه
  - بهینه‌سازی در تولید کد؛ مثل حذف محاسبات تکراری یا بیهوده
  - تولید آزمون خودکار
  - درستی‌سنجی و تضمین امنیت در نرم‌افزار
- تفاوت تحلیل ایستا و تحلیل پویا

تحلیل برنامه (ادامه)

- چالش‌های تحلیل برنامه
  - عدم محاسبه‌پذیری همه رفتارهای ممکن یک برنامه
  - تصمیم‌پذیر نبودن سوالات تحلیل
    - مثال: خاتمه‌پذیری
    - فرض کنید تابع terminate(P) همواره خاتمه می‌یابد و true برمی‌گرداند اگر و فقط اگر P به ازای تمامی داده‌های ورودی خاتمه‌پذیر باشد،

```
P ≡ while terminate(P) do skip od
```

تحلیل برنامه (ادامه)



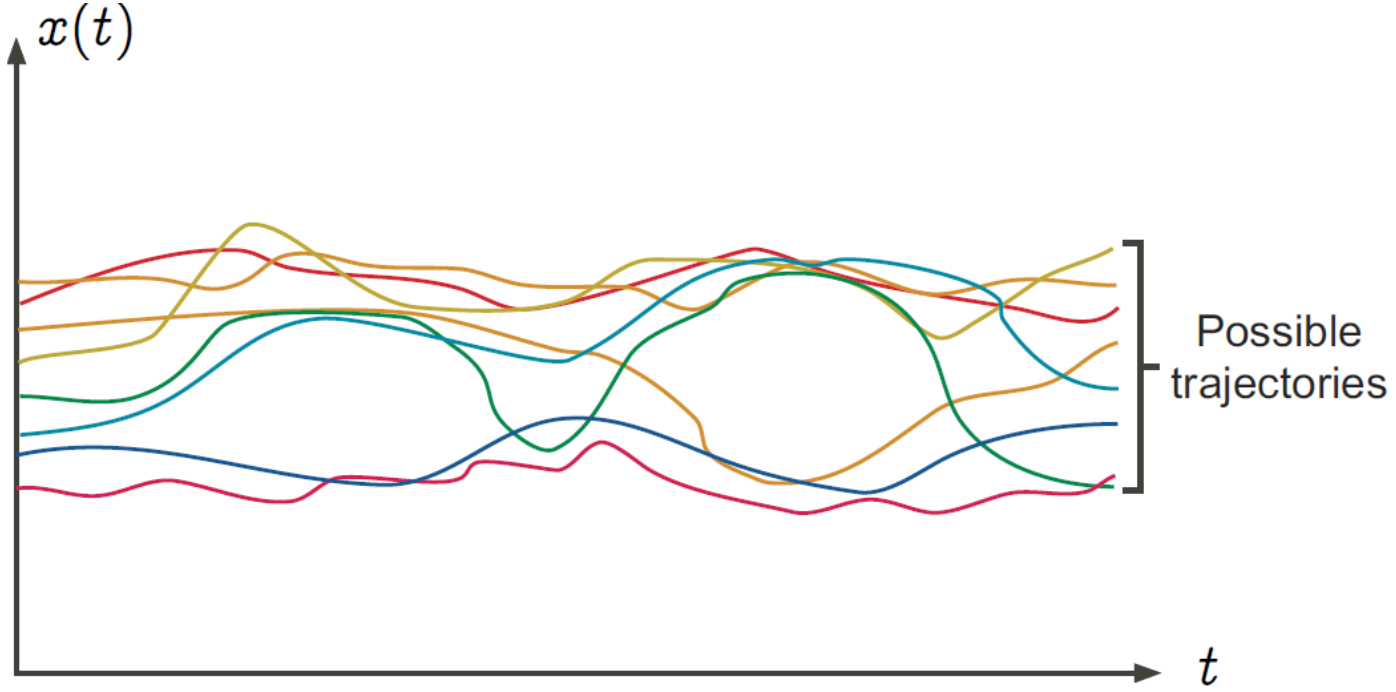
- رویکردهای تحلیل برنامه [۳]
  - تحلیل جریان داده
    - مدل‌سازی برنامه در قالب گراف جریان کنترل، جمع‌آوری اطلاعات برای هر نقطه از برنامه و حل معادلات
  - تحلیل مبتنی بر قید
    - استخراج قیدها از متن برنامه و گراف جریان کنترل، و حل آن‌ها
  - نوع‌سامانه
    - سامانه‌ای مبتنی بر منطق برای توصیف و استنتاج نوع‌ها و اثرات محاسباتی برنامه
  - تفسیر انتزاعی
    - محاسبه تحلیل با استفاده از انتزاعی از برنامه، به جای ایجاد دستگاه معادلات و سپس حل آن‌ها

تفسیر انتزاعی

- تعریف تفسیر انتزاعی
  - نظریه‌ای برای به‌دست‌آوردن تقریبی درست از رفتار برنامه‌ها [۴]
  - روشی برای طراحی معناشناخت تقریبی از برنامه‌ها که برای جمع‌آوری اطلاعات درباره برنامه‌ها و پاسخ به سوالات در خصوص رفتار زمان‌اجرای آن‌ها قابل استفاده است [۵]
  - چارچوبی نظری برای طراحی معناشناخت تقریبی درست همزمان با ایجاد آن [۶]
  - روشی برای حل یک مسئله (تصمیم‌ناپذیر) تحلیل ایستا به کمک انتزاع دامنه [۶]

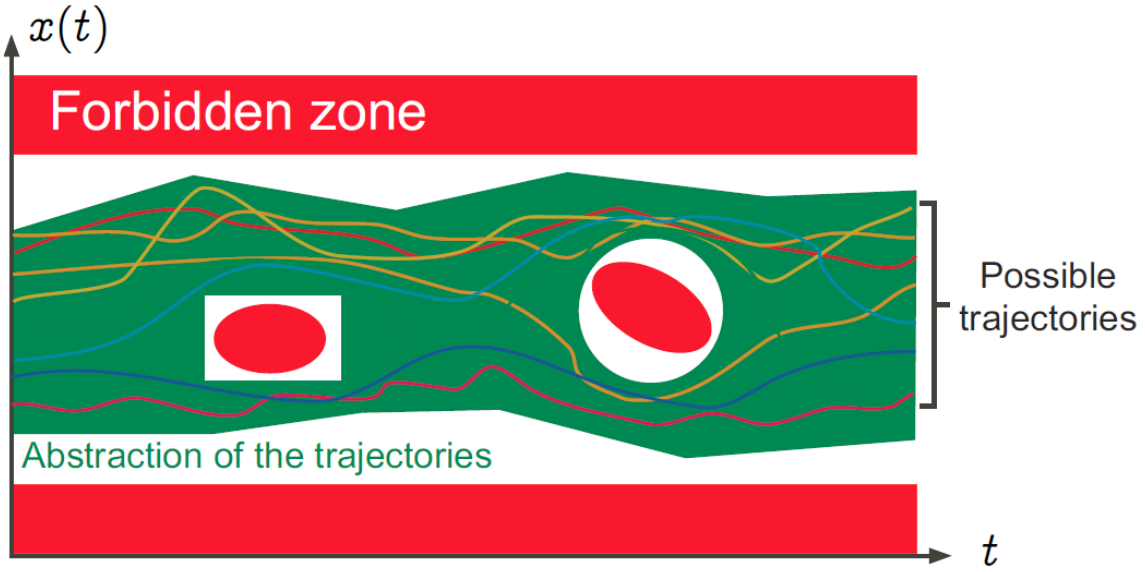
تفسیر انتزاعی (ادامه)

- معناشناخت واقعی
  - صوری‌سازی مجموعه تمام اجراهای ممکن از یک برنامه در تمامی محیط‌های ممکن اجرا (همه رفتارهای ممکن برنامه)



تفسیر انتزاعی (ادامه)

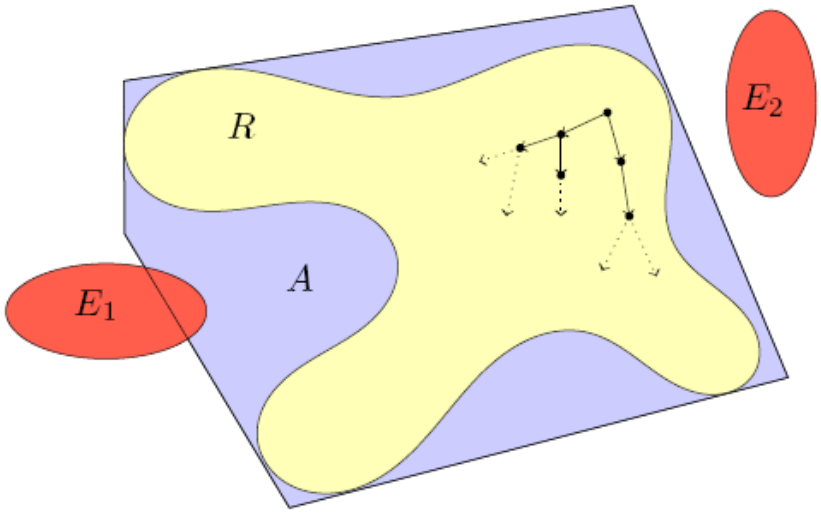
- انتزاعی از معناشناخت برنامه



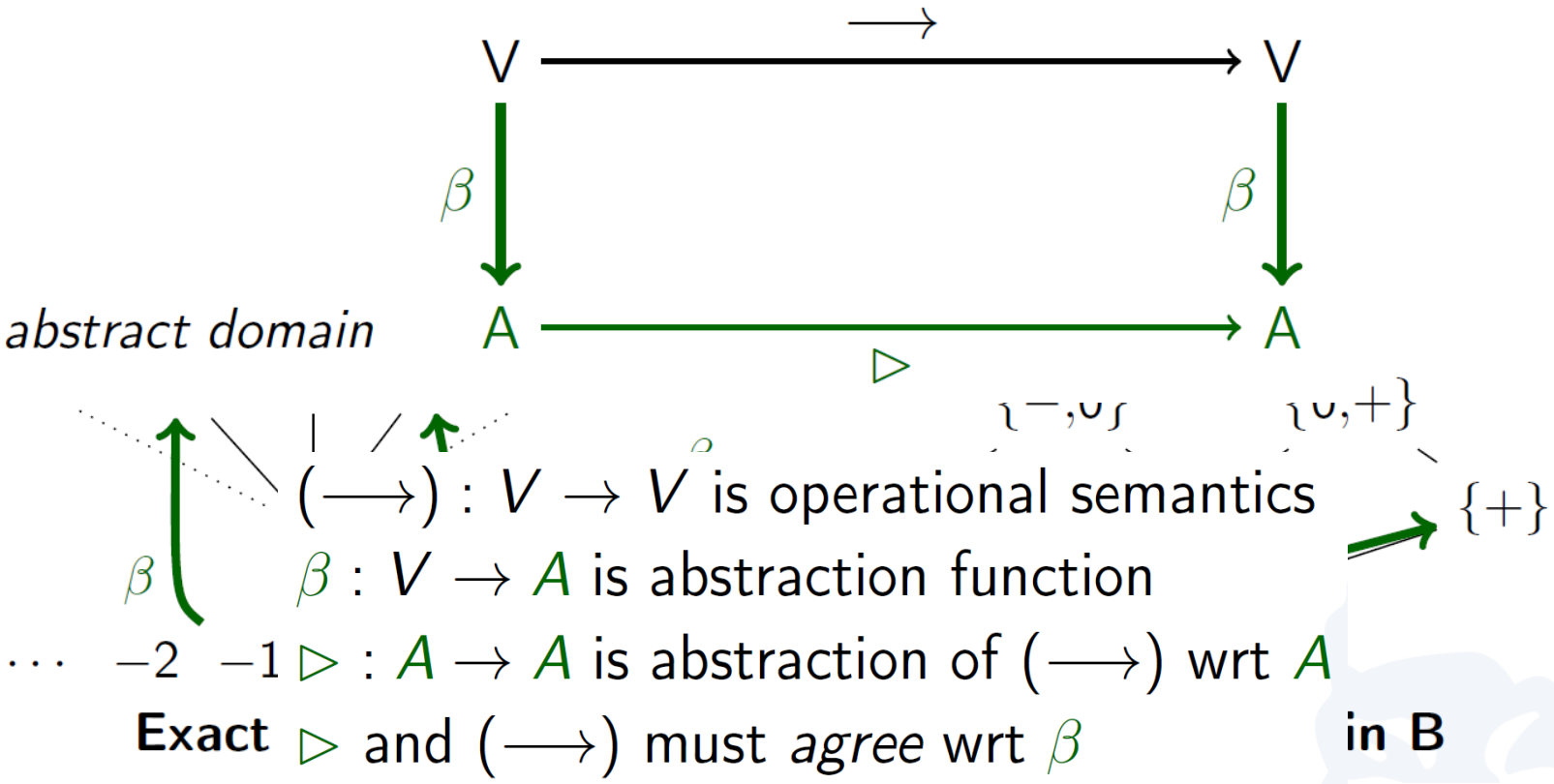
- شاخص‌های یک معناشناخت انتزاعی مطلوب
  - درستی
  - دقت
  - سادگی

دامنه انتزاعی

- زبَرمجموعه‌ای از دامنه واقعی
- چالش اصلی در تعریف دامنه انتزاعی
  - دقت تقریب دست بالا (over-approximation)
  - قابلیت بازنمایی توسط ماشین

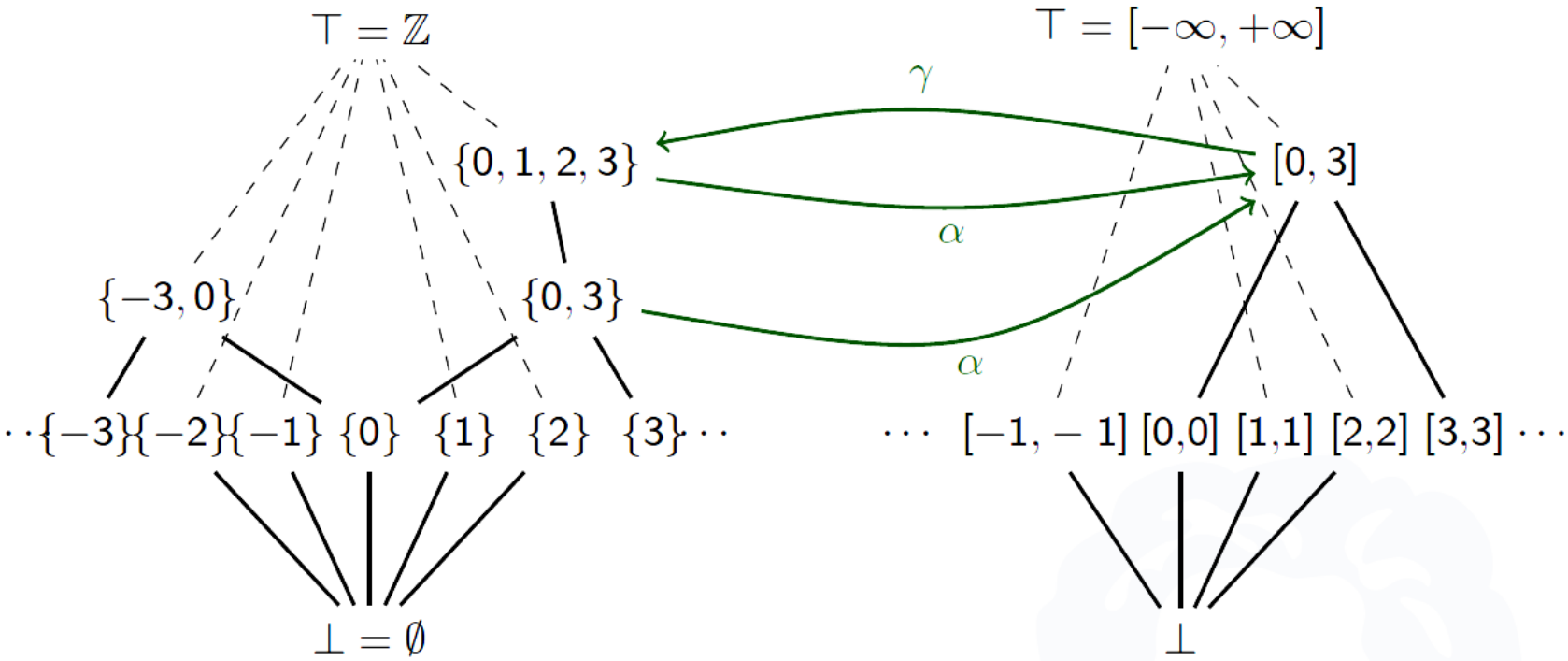


دامنه انتزاعی (ادامه)



دامنه انتزاعی (ادامه)

- مثالی از دامنه انتزاعی بازه‌ای



دامنه انتزاعی (ادامه)

- تعریف صوری

فرض کنید X دامنه واقعی و $(X, \sqsubseteq, \sqcup, \sqcap)$ یک مشبکه باشد. یک دامنه انتزاعی روی دامنه واقعی X، زوجی مانند $(X^\sharp, \gamma)$ است به طوری‌که

$\gamma: X^\sharp \to X$ (تابع واقعی‌سازی) و $\left(X^\sharp, \sqsubseteq^\sharp, \sqcup^\sharp, \sqcap^\sharp\right)$ یک مشبکه باشد که

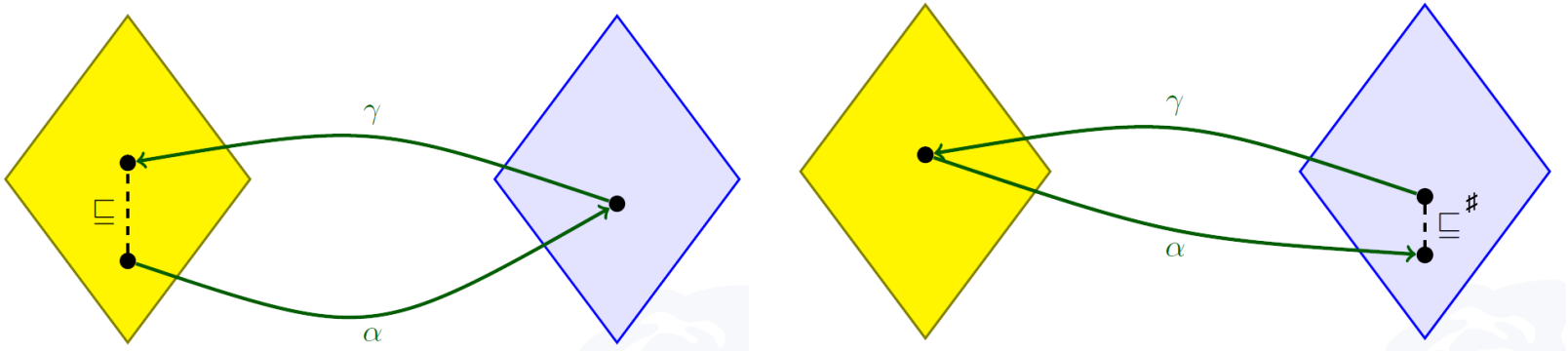
$\forall x^\sharp, y^\sharp \in X^\sharp . x^\sharp \sqsubseteq^\sharp y^\sharp \Rightarrow \gamma(x^\sharp) \sqsubseteq \gamma(y^\sharp)$ (شرط یکنوایی)

- شرط درستی انتزاع: $x \sqsubseteq \gamma(x^\sharp)$
- دقیق‌تر بودن: $x^\sharp \sqsubseteq^\sharp y^\sharp$

اتصال گالوایی

$$X \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} X^{\sharp}$$

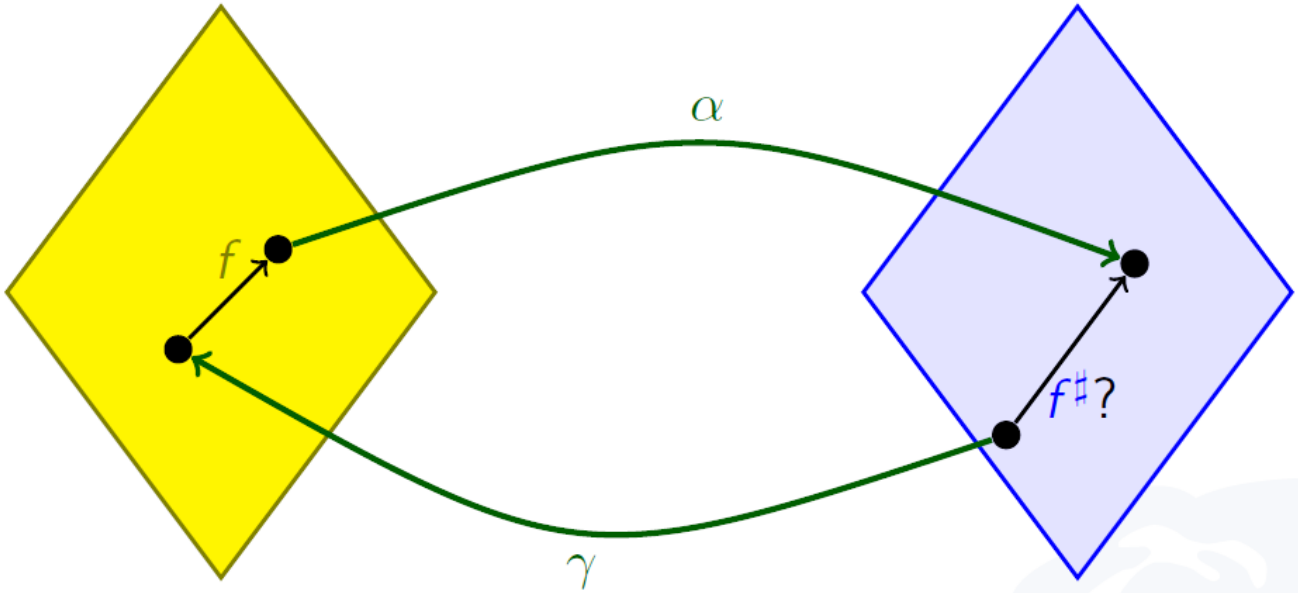
- $\alpha: X \to X^{\sharp}$ (تابع انتزاع) یکنوا باشد؛ یعنی $x \sqsubseteq y \Rightarrow \alpha(x) \sqsubseteq^{\sharp} \alpha(y)$
- $\gamma: X^{\sharp} \to X$ (تابع واقعی‌سازی) یکنوا باشد.
- $y^{\sharp} \sqsubseteq^{\sharp} \alpha \circ \gamma(y^{\sharp})$
- $\gamma \circ \alpha(x) \sqsubseteq x$



اتصال گالوایی (ادامه)

انتزاع تابع (عملیات القایی)

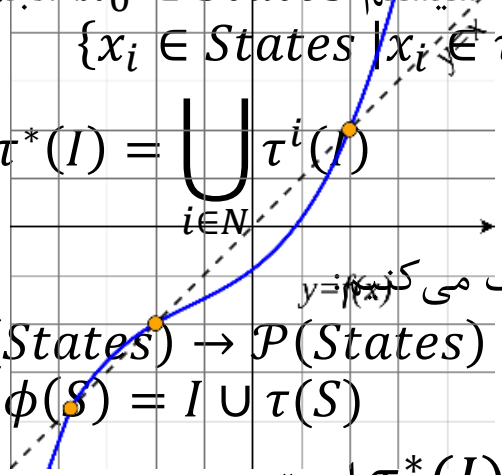
$$f^{\sharp} = \alpha \circ f \circ \gamma$$



نقطه ثابت

- با فرض مجموعه مرتب جزئی $\langle D, \sqsubseteq \rangle$ و تابع $\phi: D \to D$، یک نقطه ثابت برای تابع ϕ عضوی مانند $x \in D$ است به طوری که $\phi(x) = x$.
- در صورت تعریف تابع ϕ مناسب برای مسئله تحلیل، می‌توان مسئله را به محاسبه نقطه ثابت آن تابع تبدیل کرد.
- مثال: به دست آوردن حالت‌های قابل دسترسی برنامه
  - با شروع از یک حالت اولیه سیستم $x_0 \in States$، مجموعه حالت‌های قابل دسترسی برنامه برابر است با $\{x_i \in States \mid x_i \in \tau^i(\{x_0\})\}$

$$\tau^*(I) = \bigcup_{i \in N} \tau^i(I)$$

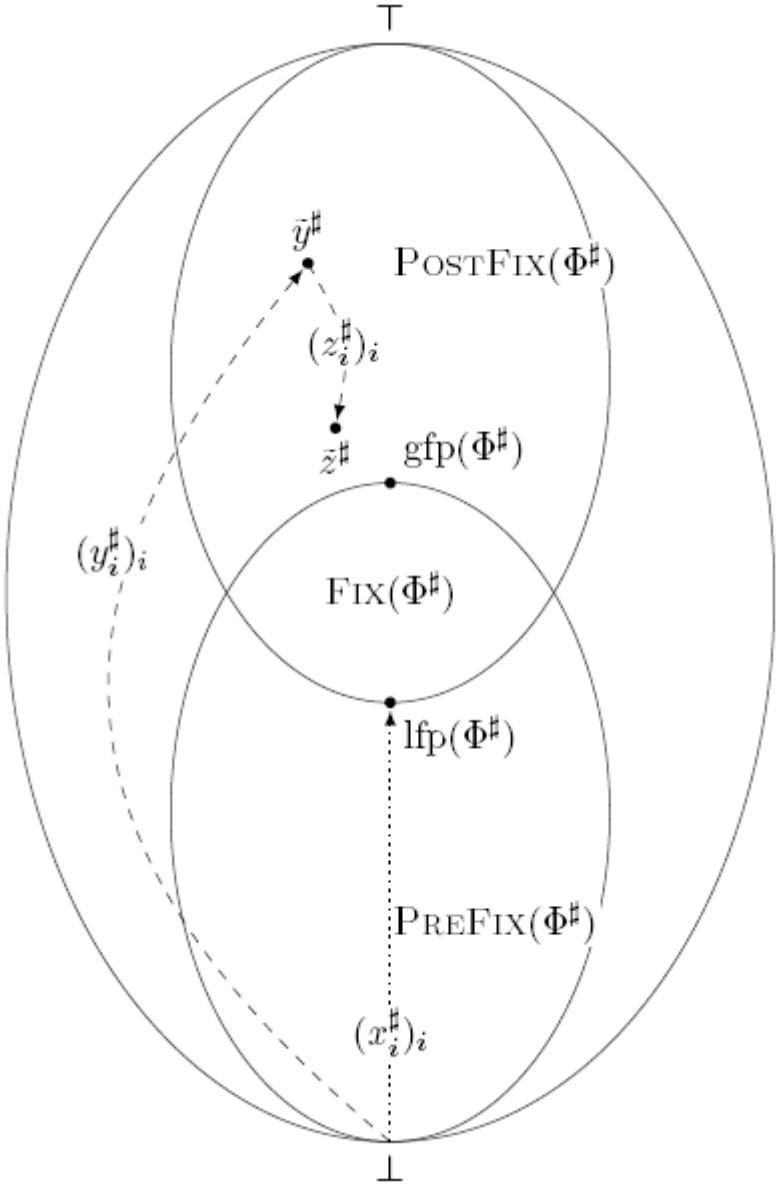


تابع ϕ را چنین تعریف می‌کنیم

$$\phi: \mathcal{P}(States) \to \mathcal{P}(States)$$
$$\phi(S) = I \cup \tau(S)$$

که نقطه ثابت آن $\tau^*(I)$ است.

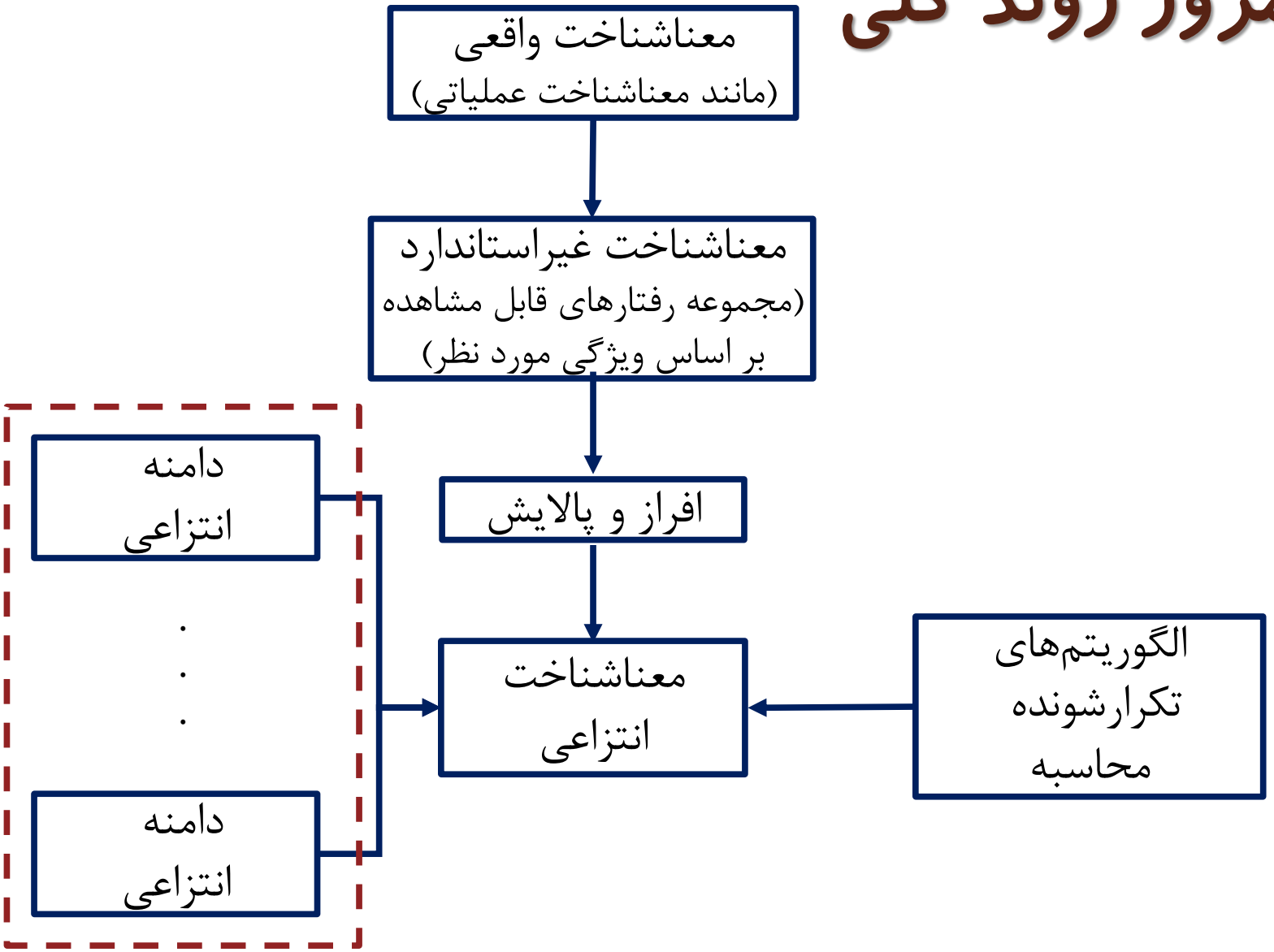
محاسبه نقطه ثابت



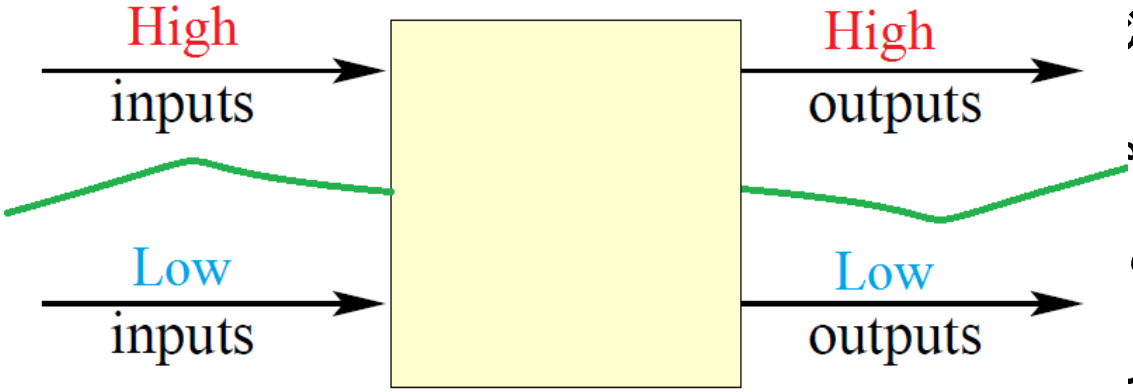
- در صورتی که تابع ϕ یکنوا با
  داد که کوچکترین نقطه ثابت
  - اما می‌توان به جای محاسبه نقط

- در صورتی که شرط زنجیره ص
  دامنه انتزاعی $X^\sharp$ برقرار نباشد
  تکرارشونده بالا پایان نمی‌یابد
  - راه حل: استفاده از عملگرهای ت

مرور روند کلی



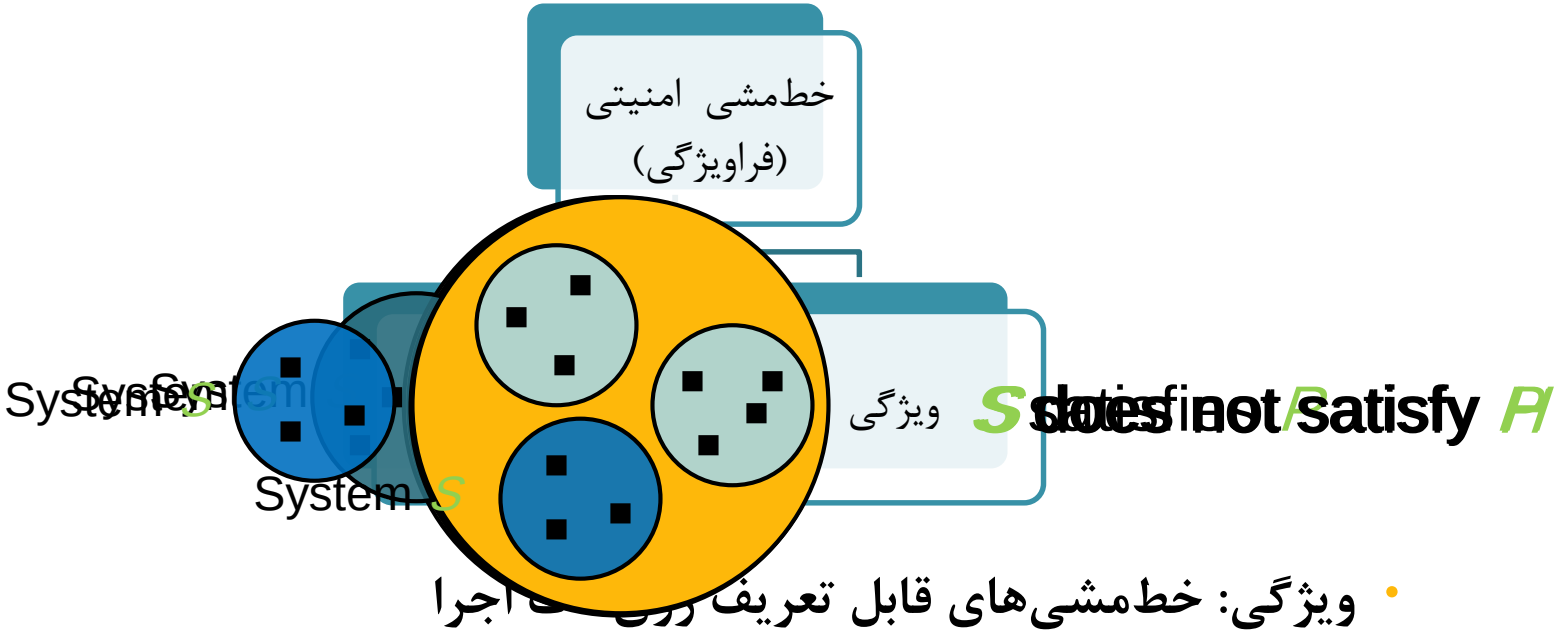
خط‌مشی‌های امنیتی



- خط‌مشی
  - محرد
- تعریف
  - مقایه
- خط‌مشی امنیتی عدم تداخل [۷]
  - بیان گزاره‌هایی روی اجراهای برنامه
  - انواع مختلف عدم تداخل

خط‌مشی امنیتی؛ ویژگی‌ها و فراویژگی‌ها

- تعریف صوری خط‌مشی امنیتی



- ویژگی: خط‌مشی‌های قابل تعریف روی یک اجرا
  - مانند عدم خاتمه
- فراویژگی [۸]: خط‌مشی‌های قابل تعریف روی مجموعه‌ای از اجراها
  - ناویژگی مانند جریان اطلاعات
    - trace
    - = trace

به‌کارگیری تفسیر انتزاعی در اعمال ویژگی‌ها و فراویژگی‌ها

- ویژگی‌ها
  - **ایمنی**: چیز بد هیچ‌گاه رخ نمی‌دهد
    - نامتغیر: ویژگی‌ای که باید در هر حالت از تمامی اجراهای برنامه برقرار باشد [۶]
  - مشخص‌بودن به‌کارگیری تفسیر انتزاعی برای ویژگی‌های ایمنی
    - بررسی عدم اشتراک معناشناخت انتزاعی با نواحی ممنوعه با توجه به ویژگی ایمنی
    - تبدیل به نزدیک‌ترین نامتغیر، در صورت عدم امکان تحلیل ویژگی ایمنی
  - **مانایی**: چیز خوب بالاخره اتفاق می‌افتد [۹، ۱۰]
    - تحلیل ایستای خاتمه‌پذیری [۱۱]
    - تحلیل ایستای فرمول‌های منطق زمانی CTL [۱۲]

به‌کارگیری تفسیر انتزاعی در اعمال ویژگی‌ها و فراویژگی‌ها (ادامه)

- ناویژگی‌ها (نیازمند بررسی ارتباط بین چند اجرا از برنامه)
  - رویکردی برای تحلیل ایستای۲-ایمنی‌ها با استفاده از تفسیر انتزاعی رابطه‌ای روی خودترکیبی گراف‌های جریان کنترل برنامه [۱۳]
    - فقط عدم تداخل غیرحساس به خاتمه
  - به‌کارگیری تفسیر انتزاعی برای نظارت عدم تداخل به کمک منطق رابطه‌ای [۱۴]
    - محاسبه اجراهای فرعی سازگار با اجرای فعلی در هر وضعیت ناظر و بررسی ویژگی ایمنی برای اجراهای فرعی

به‌کارگیری تفسیر انتزاعی در اعمال ویژگی‌ها و فراویژگی‌ها (ادامه)

- ناویژگی‌ها
  - تکنیک روش‌مند برای طراحی ناظر مبتنی بر تفسیر انتزاعی برای خط‌مشی‌ها با در نظر گرفتن تنزل سطح [۱۵]
    - صوری‌سازی تعبیر ناظر ایده‌آل به کمک اتصال گالوایی
    - ناظر درست همگام با ساخت
  - بیان و اثبات درستی تحلیل ایستای جریان اطلاعات امن در قالب تفسیر انتزاعی [۱۶]
    - با تعریف یک اتصال گالوایی که برای تقریب مستقیم فراویژگی
    - تعریف مجموعه‌ای از مجموعه تبدیل‌کننده‌ها
    - دربرگرفتن فراایمنی‌هایی که k-ایمنی نیستند
    - ارائه تحلیل برای جریان اطلاعات کمّی

بعضی از ابزارهای مبتنی بر تفسیر انتزاعی

- Airac5: تحلیل‌گر ایستا برای تشخیص خودکار خطاهای سرریز بافر در برنامه‌های زبان C
- تحلیل‌گرهای aiT WCET: تحلیل‌گرهای ایستای تجاری برای محاسبه ایستای محدوده بدترین حالت زمان اجرای برنامه در سامانه‌های بی‌درنگ
- ASTREE: تحلیل‌گر ایستا برای اثبات عدم وجود خطاهای زمان‌اجرا، مخصوص برنامه‌های کنترل همگام در زبان C
- C Global Surveyor: تحلیل‌گر ایستای نمونه‌اولیه برای یافتن خطاهای زمان‌اجرا در برنامه‌های زبان C
- Fluctuat: تحلیل‌گر ایستا برای مطالعه انتشار خطاهای رندکردن محاسبات ممیز شناور در برنامه‌های C یا اسمبلی
- PolySpace Verifier: تحلیل‌گر ایستای تجاری و عام‌منظوره برای تشخیص خطاهای زمان‌اجرا در زبان‌های Ada، C، C++ و Java
- TERMINATOR: تحلیل‌گر ایستا برای اثبات خاتمه برنامه و ویژگی‌های مانایی در زبان C

چالش‌ها و مسائل باز

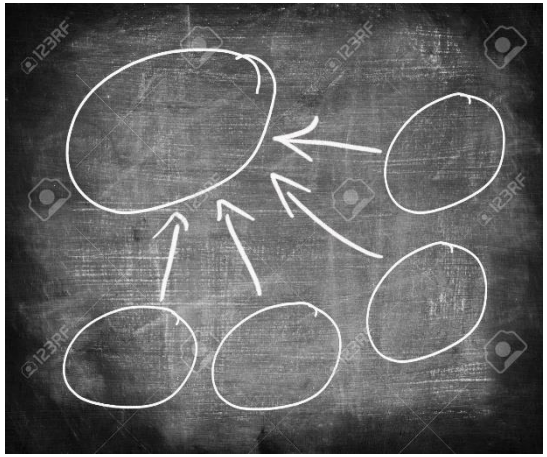
- انتخاب دامنه‌های انتزاعی به جز اعداد حسابی
  - آرایه‌ها، حافظه هیپ و نوع‌های دیگر
- به‌کارگیری تفسیر انتزاعی برای برنامه‌های هم‌روند و ویژگی‌های احتمالاتی
- تحلیل ویژگی‌های زمانی و مانایی با در نظر گرفتن شرط‌های انصاف
- تحلیل ویژگی‌های LTL و CTL*
- پشتیبانی از فراخوانی توابع و مفاهیم شی‌گرایی
- در نظر گرفتن عدم تداخل حساس به خاتمه و زمان
- ارائه چارچوب پارامتریک برای تحلیل ایستای ناویژگی‌ها و افزایش دقت آن

چالش‌ها و مسائل باز

- بررسی الگوریتم‌های تکرارشونده برای محاسبه نقطه‌ثابت در تحلیل ناویژگی‌ها
- سرشت‌نمایی و برقراری نگاشت بین تحلیل‌های ایستا و تفسیر انتزاعی
- ارزیابی و بررسی توانایی دقت ناظرهای ساخته‌شده به کمک تفسیر انتزاعی
- ساخت ناظر مبتنی بر تفسیر انتزاعی برای زبان‌های واقعی
- تجمیع تحلیل مبتنی بر تفسیر انتزاعی با چرخه توسعه نرم‌افزار

جمع‌بندی

- مروری بر مفاهیم اولیه تحلیل برنامه
- معرفی تفسیر انتزاعی و انواع دامنه‌های انتزاعی
- تعریف خط‌مشی جریان اطلاعات به عنوان یک ناویژگی
- مروری بر روش‌های تحلیل ایستای مبتنی بر تفسیر انتزاعی برای ویژگی‌ها و فراویژگی‌ها
- بیان چالش‌ها و مسائل باز حوزه



لیست کنفرانس‌ها و مجله‌ها مرتبط

- کنفرانس‌ها

- Verification, Model Checking, and Abstract Interpretation (VMCAI) – since 1999
- Computer-Aided Verification (CAV) – since 1989
- Static Analysis Symposium (SAS) – since 1991
- Principles of Programming Languages (POPL) – since 1973
- Computer Security Foundations Symposium (CSF) – since 1988
- Conference on Computer and Communications Security (CCS) – since 1993
- European Symposium on Programming (ESOP) – since 1991

- مجله‌ها

- Journal of Logic and Computation – since 1990
- Transactions on Programming Languages and Systems – since 1979

منابع و مراجع

[1] B. Christel and J. P. Katoen. "Principles of Model Checking." *MIT press*, 2008.

[2] P. Cousot and R. Cousot. "A Gentle Introduction to Formal Verification of Computer Systems by Abstract Interpretation." In *Logics and Languages for Reliability and Security*, volume 25 of NATO Science for Peace and Security Series - D: Information and Communication Security, pp. 1–29. IOS Press, 2010.

[3] F. Nielson, H. R. Nielson, and C. Hankin. "Principles of Program Analysis." *Springer*, 1999.

[4] P. Cousot. "Program Analysis: The Abstract Interpretation Perspective." *ACM Computing Surveys (CSUR)* - Special issue: position statements on strategic directions in computing research, vol. 28, pp. 73-76, 1996.

[5] P. Cousot and R. Cousot. "Abstract Interpretation Frameworks." in *Journal of Logic and Computation*, pp. 511–547, 1992.

[6] J. Henry, "Static Analysis by Abstract Interpretation and Decision Procedures." PhD Thesis, Université de Grenoble, 2014.

[7] J. A. Goguen and J. Meseguer, "Security Policies and Security Models," in *IEEE Symposium on Security and Privacy*, pp. 11-24, 1982.

[8] M. R. Clarkson and F. B. Schneider. "Hyperproperties." in *Journal of Computer Security,* vol. 18, no. 6 pp. 1157-1210, 2010.

[9] C. Urban. "Static Analysis by Abstract Interpretation of Functional Temporal Properties of Programs." PhD Thesis, Ecole normale supérieure - ENS PARIS, 2015.

منابع و مراجع

[10] D. Masse. "Property Checking Driven Abstract Interpretation-Based Static Analysis." In *Verification, Model Checking, and Abstract Interpretation (VMCAI)*, pp. 56–69, 2003.

[11] P. Cousot and R. Cousot. "An Abstract Interpretation Framework for Termination." In *Principles of Programming Languages (POPL)*, pp. 245–258, 2012.

[12] C. Urban, S. Ueltschi, and P. Muller. "Abstract Interpretation of CTL Properties." In *Static Analysis Symposium (SAS)*, pp. 4-24, 2018.

[13] M. Kovács, H. Seidl, and B. Finkbeiner. "Relational Abstract Interpretation for the Verification of 2-hypersafety Properties." In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 211-22, 2013.

[14] A. Chudnov, G. Kuan, and D.A. Naumann. "Information Flow Monitoring as Abstract Interpretation for Relational Logic." In *IEEE Computer Security Foundations Symposium (CSF)*, pp. 48-62, 2014.

[15] M. Assaf, and D.A. Naumann. "Calculational Design of Information Flow Monitors." In *IEEE Computer Security Foundations Symposium (CSF)*, pp. 210-224, 2016.

[16] M. Assaf,, D.A. Naumann, J. Signoles, E. Totel, and F. Tronel. "Hypercollecting Semantics and Its Application to Static Analysis of Information Flow." In *ACM SIGPLAN Notices* ,vol. 52, no. 1, pp. 874-887, 2017.

[17] P. Cousot. MIT Course 16.399: Abstract Interpretation. http://web.mit.edu/afs/athena.mit.edu/course/16/16.399/www/, 2005.

[18] C. Reichenbach, Goethe University Frankfurt Course, Foundations of Programming Languages, http://www.sepl.informatik.uni-frankfurt.de/2014-ws/m-ps/index.en.html, 2014.

با سپاس از توجه شما!