روشی مبتنی بر تطبیق الگو برای تخمین بیشترین زمان اجرای حلقه‌های یکنواخت چندمسیری

مهدی سخائی‌نیا و سعید پارسا

چکیده: روش تطبیق الگو یکی از روش‌هایی است که برای تخمین بیشترین زمان اجرای حلقه‌ها ارائه شده است. در این روش در صورتی که حلقه با الگوی ارائه‌شده تطبیق داشت با استفاده از یک معادله، بیشترین تعداد تکرار حلقه محاسبه می‌گردد. در حقیقت برای محاسبه تعداد تکرار نیازی نیست که مقدار متغیرهای کنترلی حلقه برای هر تکرار محاسبه گردد. نقص روش تطبیق الگو وابستگی زیاد آن به الگو است. این وابستگی به ساختار و محل شرط تست‌کننده متغیر کنترلی حلقه و از سوی دیگر به محل، نحوه و تعدد تغییر متغیر کنترلی حلقه مرتبط است. برای کاهش وابستگی به الگو می‌توان جریان اطلاعات برای حلقه‌های یکنواخت چندمسیری در قالب دو دسته عبارت نمادین، نشان‌دهنده شرط تکرار و نحوه تغییر متغیرهای کنترلی حلقه را مدل‌سازی کرد. بر اساس این عبارات، تعداد مقادیر ممکن که در زمان اجرا می‌توان به متغیرهای کنترلی حلقه تخصیص داد محاسبه و به عنوان تخمینی از بیشترین تعداد تکرار ارائه می‌گردد. اما تخمین ارائه‌شده در این روش بیشتر از مقدار واقعی است و در اصطلاح دارای بیش‌تخمین خواهد بود. در این مقاله، متغیرهایی که مقدارشان در مسیرهای تکرار مختلف یکسان هستند و در هر چند مسیر این مقدار به عنوان یک تکرار محاسبه گردیده است، شناسایی و در محاسبه‌ها لحاظ می‌گردند. این کار باعث می‌گردد که مقدار بیش‌تخمین کاهش یابد. ارزیابی‌ها نشان داد که روش ارائه‌شده در این مقاله روشی مؤثر و کارا بوده و بیش‌تخمین کمتری دارد.

کلیدواژه: تخمین بیشترین زمان اجرا، تحلیل حد حلقه‌های تکرار، سامانه‌های نهفته بی‌درنگ، تحلیل ایستای برنامه.

۱- مقدمه

در سامانه‌های بی‌درنگ برای یافتن یک زمان‌بندی ممکن و اطمینان از خاتمه یک وظیفه[1] در مهلت[2] تعیین‌شده، زمان‌بند باید مدت زمان اجرای هر وظیفه را بداند. محاسبه دقیق مدت زمان اجرای یک وظیفه ممکن نیست چون برنامه یا وظیفه در هر اجرا بر اساس ورودی‌ها از مسیرهای متفاوتی اجرا شده که زمان اجرای دستورها در این مسیرها متفاوت است. بنابراین مدت زمان اجرای یک وظیفه در اجراهای مختلف آن بر روی یک بستر مشخص متفاوت است. بر این اساس برای یافتن یک زمان‌بندی درست و مطمئن، بیشترین زمان اجرای (WCET)[3] هر وظیفه باید تخمین زده شود تا تضمین گردد که در این زمان‌بندی مهلت‌ها برآورده خواهد شد [۱].

بیشتر نرم‌افزارهای سامانه‌های نهفته بی‌درنگ مشتمل بر حلقه‌های تکرار زیادی هستند. تخمین غیر دقیقی از تعداد تکرار این کدها حتی به میزان کمی بر بیشترین زمان اجرای تخمینی برنامه تأثیر به‌سزایی دارد. از همین روست که تخمین مطمئن و نزدیک به مقدار واقعی بیشترین زمان اجرا برای حلقه‌های تکرار از اهمیت ویژه‌ای برخوردار است [۲].

یکی از روش‌های تخمین بیشترین زمان اجرای حلقه‌های تکرار روش تطبیق الگو[4] است [۱]. روش تطبیق الگو بر پایه این واقعیت است که برای بیشتر حلقه‌های تکرار، مترجم‌ها[5] از مجموعه دستورهای ماشین یکسان یا مشابه برای مقداردهی، به روز رسانی و تست شمارنده حلقه استفاده می‌نمایند. روش‌های تطبیق الگو این گونه دستورها را در کد برنامه یافته و مقادیر عملوندهای این دستورها را تحلیل نموده تا بازه مقادیری که شمارنده می‌پذیرد را محاسبه نمایند. مقادیری که می‌توان به شمارنده انتساب داد به طور معمول در قالب مقدار اولیه، میزان افزایش یا کاهش و مقدار نهایی شمارنده بیان می‌گردد. سپس می‌توان با استفاده از یک معادله بلافاصله تعداد تکرار حلقه را محاسبه نمود. نقص روش‌های تطبیق الگو این است که اگر بهینه‌سازی مترجم انجام گردد یا مترجم خودش تغییراتی داشته باشد، تطبیق الگو شکست خورده و نمی‌تواند تعداد تکرار را به درستی محاسبه نماید. به طور کلی عدم تطبیق با الگو سبب شکست این روش‌ها می‌گردد [۱]. سایر روش‌های محاسبه تکرار که مبتنی بر تطبیق الگو نیستند باید مقدار شمارنده را در هر تکرار محاسبه نمایند و سپس تعداد تکرار را به دست آورند که زمان اجرای ابزار برای تحلیل بزرگ خواهد بود و کارایی کمی خواهد داشت [۳] و [۴].

الگو در روش‌های تطبیق الگو وابسته به دو مجموعه دستورها خواهد بود. اولین مجموعه مربوط به دستورهایی است که شمارنده را مقداردهی و به روز رسانی می‌کنند یا در اصطلاح چگونگی تغییر مقدار شمارنده را نشان می‌دهند. دومین مجموعه دستورها مربوط به تست شمارنده است. این دستورها در غالب یک عبارت شرطی است که می‌توان گفت شرط تکرار و یا خاتمه یک حلقه هستند. این روش‌ها اغلب محدود به موارد زیر هستند:

- ساختار و محل شرط تکرار و خاتمه
- محل و چگونگی تغییر شمارنده در بدنه حلقه تکرار
- تعداد شمارنده‌ها

در [۵] روش نوینی ارائه کردیم که اثرپذیری کمتری از محدودیت‌های فوق داشته است. برای حل این مشکل در این مقاله اطلاعات جریان کنترلی و داده‌ای حلقه تکرار استخراج گردیده و در قالب دو دسته عبارت نمادین مدل شده است. بدین منظور مسیرهای تکرار در بدنه حلقه که در

این مقاله در تاریخ ۵ خرداد ماه ۱۳۹۷ دریافت و در تاریخ ۴ بهمن ماه ۱۳۹۷ بازنگری شد.

مهدی سخائی‌نیا (نویسنده مسئول)، گروه مهندسی کامپیوتر، دانشکده مهندسی، دانشگاه بوعلی سینا، همدان، ایران، (email: sakhaei@basu.ac.ir).

سعید پارسا، گروه مهندسی نرم‌افزار، دانشکده کامپیوتر، دانشگاه علم و صنعت ایران، تهران، ایران، (email: parsa@iust.ac.ir).

1. Task
2. Deadline
3. Worst-Case Execution Time
4. Pattern-Matching
5. Compiler

هر تکرار، حلقه از یکی از آنها اجرا می‌شود شناسایی می‌گردد. برای هر مسیر تکرار یک عبارت شرط تکرار و عبارات چگونگی تغییر شمارنده‌ها ایجاد می‌گردد. عبارت شرط مسیر تکرار نشان می‌دهد با چه مقادیری از شمارنده‌ها اجرای حلقه بر روی این مسیر خواهد بود و عبارت چگونگی تغییر شمارنده نشان می‌دهد که در این مسیر به مقدار هر یک از شمارنده‌ها چه مقداری اضافه یا کم می‌گردد. در هر تکرار یک مجموعه مقدار برای شمارنده‌ها وجود دارد که سبب ارزیابی مقدار شرط تکرار به درست[1] می‌گردد. برای حلقه‌های چندمسیری[2] که تغییر شمارنده یکنواخت[3] است به کمک شرط تکرار برای هر مسیر، تعداد ممکن این مجموعه، محاسبه و به عنوان بیشینه تعداد تکرار بر روی این مسیر لحاظ می‌گردد اما مقدار محاسبه‌شده دارای بیش‌تخمین است.

در مقاله حاضر روش ارائه‌شده در [۵] توسعه داده شده است[4] به نحوی که بیش‌تخمین ذکرشده کاهش پیدا نموده و بیشترین زمان اجرای تخمینی به مقدار واقعی آن نزدیک گردد. همان گونه که ذکر گردید به کمک شرط تکرار برای هر مسیر، تعداد ممکن مجموعه مقادیری که شرط تکرار را درست ارزیابی می‌کند محاسبه می‌گردد و به عنوان بیشینه تعداد تکرار بر روی این مسیر لحاظ می‌گردد. اما همه مقادیری که منجر به ارزیابی صحیح شرط تکرار می‌گردند در اجرای واقعی رخ نمی‌دهند. با استفاده از عبارت نحوه تغییر شمارنده این حالات شناسایی و از بیشینه تعداد مقادیر تکرار کم می‌گردد. همچنین مجموعه مقادیر یک متغیر بین دو یا چندمسیر تکراری ممکن است مشترک باشد که چون برای هر دو مسیر به عنوان تعداد تکرار آن محسوب می‌گردد سبب ایجاد بیش‌تخمین می‌شود. با در نظر گرفتن این مشکل در روش ارائه‌شده سعی گردیده که بیش‌تخمین کاهش یابد.

در ادامه مقاله در بخش دوم به کارهای مرتبط پرداخته و در بخش سوم روش پیشنهادی با جزئیات و مثال تشریح شده است. توانایی روش ارائه‌شده در بخش چهارم ارزیابی گردیده و در نهایت در بخش پنجم نتیجه‌گیری و کارهای آینده شرح داده شده است.

۲- کارهای مرتبط

روش‌های مختلفی برای تخمین تعداد تکرارهای حلقه به صورت خودکار مطرح گردیده‌اند [۱]. در روش‌های مبتنی بر تطبیق الگو برای حلقه‌های تکرار بر اساس شمارنده، معمولاً با یافتن مقدار اولیه و نهایی شمارنده و مشخص‌نمودن چگونگی تغییر شمارنده، تعداد تکرار حلقه محاسبه می‌گردد. توانمندی و ضعف روش‌های ارائه‌شده وابسته به محدودیت‌هایی است که برای ساختار حلقه در نظر گرفته‌اند.

در روش ارائه‌شده در [۶] تعداد تکرار حلقه برای حلقه‌ها با چند نقطه خروج و حلقه‌های تودرتوی مثلثی تعیین گردیده است. در این روش تعداد تکرار در قالب یک عبارت مجموع‌یابی[5] فرمول‌بندی[6] گردیده و با حل این عبارت تعداد تکرار محاسبه شده که این حلقه‌های تکرار باید با الگوی مطرح‌شده در مقاله تطبیق داشته باشد. مقدار اولیه و نهایی شمارنده باید ثابت باشد یعنی حلقه حساس به متن[7] نخواهد بود. به عبارت دیگر در فراخوانی‌های متفاوت از اجرای حلقه تعداد تکرار ثابت است و به پارامترهای ورودی روال وابسته نیست. در این روش شمارنده فقط می‌تواند با مقدار ثابت افزایش یا کاهش یابد و عملگرهای منطقی AND و OR در شرط تکرار نمی‌توانند استفاده گردند.

در ابزار Bound-T با تخمین بازه شمارنده و چگونگی افزایش آن تعداد تکرار به کمک محاسبات پرسبرگر صورت پذیرفته است [۷]. محاسبه تعداد تکرار برای بدنه حلقه و حساس به متن است. در این روش فقط امکان افزایش و کاهش شمارنده با مقدار ثابت وجود دارد [۸].

در ابزار TuBound برای محاسبه تعداد تکرار حلقه‌ها یک معادله نمادین ایجاد می‌کند که با تعدادی قانون آن را حل می‌نماید [۹]. شمارنده می‌تواند با یک متغیر (مقدار نامعلوم) مقداردهی گردد البته به شرط این که در شرط تکرار که شمارنده با یک عبارت محاسباتی مقایسه می‌گردد، در این عبارت مقدار ثابتی به آن متغیر افزوده یا کم گردد. محدودیت دیگر روش این است که در بدنه حلقه تکرار شمارنده نمی‌تواند تغییر نماید.

ابزار aiT نحوه تغییر شمارنده‌ها را از متن دستورها استخراج نموده است [۱۰]. فقط یک نقطه خروج با یک شرط می‌تواند وجود داشته باشد و شرط تکرار مقایسه یک ثابت با یک ثبّات[8] است. همچنین به شمارنده در هر تکرار فقط یک مقدار ثابت افزوده شود. در ادامه در این ابزار برای محاسبه تعداد تکرار برای حلقه‌های پیچیده‌تر یک روش مبتنی بر تحلیل جریان داده ارائه گردید. در این روش با شناسایی شمارنده و تحلیل جریان داده نحوه تغییر شمارنده را در هر نقطه از حلقه نسبت به ابتدای حلقه به دست می‌آورد. سپس با بررسی شرط خروج برای هر نقطه خروج از حلقه و همچنین نحوه افزوده‌شدن شمارنده نسبت به ابتدای حلقه تعداد تکرار حلقه از هر نقطه خروج محاسبه شده و در نهایت به صورت یک بازه بیشترین تعداد تکرار بیان می‌گردد. امکان افزودن یک غیر ثابت یا استفاده از ضرب برای تغییر شمارنده وجود ندارد. تغییر شمارنده در مسیرهای مختلف تکرار به مقدار ثابتی که به شمارنده افزوده محدود گردیده است [۱۱].

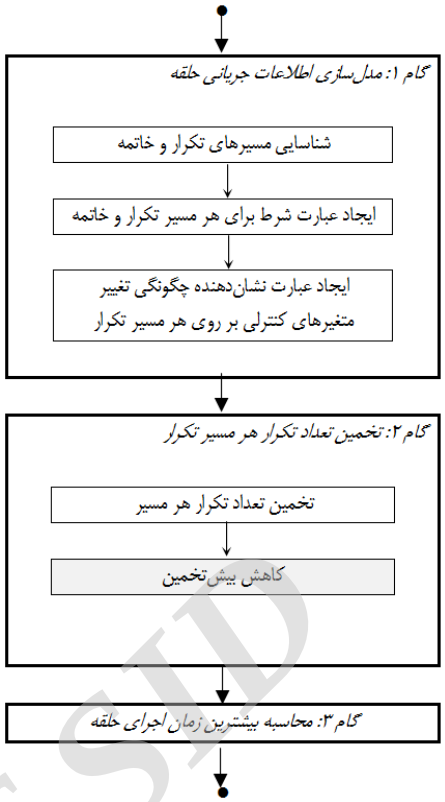
در [۱۲] حلقه‌ها با چند مسیر اجرایی در بدنه حلقه در کد منبع به حلقه‌ای با یک مسیر اجرایی تبدیل می‌گردد. در این گونه حلقه‌ها در مسیرهای اجرایی مختلف، تغییر شمارنده متفاوت است. البته مسیرها باید فقط از طریق دستورهای شرطی یک‌سطحی (بدون شرط تودرتو) تشکیل شده باشد. برای حلقه‌های با یک مسیر اجرایی یک رابطه بازگشتی ایجاد شده که با حل آن تعداد تکرار محاسبه می‌گردد. امکان نامعلوم‌بودن مقدار شروع شمارنده در این روش وجود دارد و نحوه تغییر شمارنده باید یکنواخت و خطی باشد. مقدار محاسبه‌شده، تعداد تکرار کل بدنه حلقه تکرار خواهد بود که منجر به بیش‌تخمین می‌گردد. برای حل بیش‌تخمین در [۱۳] تلاش شده که مسیرهایی را که اجرا نمی‌گردند شناسایی کرده و آنها را در طی تخمین نادیده می‌گیرد.

همان طور که ذکر شد تمامی روش‌های مبتنی بر تطبیق الگو با اعمال محدودیت‌هایی در ساختار حلقه، بیشترین زمان اجرای آن را تخمین می‌زنند. در روش ارائه‌شده در [۵] این محدودیت‌ها کمتر بوده و تخمین دقیق‌تری را ارائه می‌نماید. در این روش اگر شرط تکرار برای مسیرها در

1. True
2. Multi-Path
3. Monotonic

۴. روش ارائه‌شده در این مقاله نسبت به [۵] تخمین دقیق‌تری ارائه می‌نماید که به طور مشخص در مثال و ارزیابی این موضوع نشان داده شده و این بهبود در بخش ۳-۵ مقاله تبیین گردیده است. همچنین بخش قضیه ۳-۴ به این مقاله، افزوده و کارهای مرتبط توسعه داده شده است. برای تبیین بهتر روش ارائه‌شده بیشتر بخش‌های مقاله بازنویسی و کامل‌تر گردیده است.

5. Summation
6. Formulation
7. Context-Sensitive
8. Register



شکل ۱: نمودار عملکرد روش پیشنهادی.

بدنه حلقه تکرار یک چندوجهی بسته است، تعداد نقاط درون این چندوجهی را به عنوان بیشترین تکرار این مسیر در نظر می‌گیرد. هر نقطه نشان‌دهنده یک چندتایی از مقدار متغیرهایی است که منجر به ارزیابی عبارت شرط تکرار برای آن مسیر به مقدار درست می‌گردد. حلقه‌ها با چند مسیر و چند شمارنده در روش ارائه می‌تواند تحلیل گردند. محدودیتی در محل دستورهای شرط و تغییردهنده شمارنده و تعدد آنها وجود ندارد. محدودیت اعمال‌شده در این روش شرط تکرار برای هر مسیر باید خطی و یک چندوجهی بسته باشد و همچنین تغییر متغیر کنترلی باید یکنواخت باشد که این محدودیت‌ها در تمامی روش‌ها وجود دارد. به حساب آوردن تمامی نقاط درون چندوجهی برای تخمین تعداد تکرار سبب یک بیش‌تخمین می‌گردد. گرچه تلاش گردیده بود که با به حساب آوردن نحوه تغییر متغیرها بر روی هر مسیر میزان بیش‌تخمین کاهش یابد اما باز هم این روش دارای بیش‌تخمین هست.

در این مقاله با توسعه روش مطرح‌شده در [۵] و با لحاظنمودن مقدار متغیرهایی که در دو یا چند مسیر تکرار یکسان هستند، این بیش‌تخمین به صورت مؤثرتری کاهش یافته است.

۳- روش پیشنهادی

برای تخمین بیشترین زمان اجرای یک حلقه، تعداد دفعات تکرار اجزای حلقه (مسیر تکرار) و بیشترین زمان اجرای هر جزء محاسبه شده و بر اساس آن بیشترین زمان اجرای کل حلقه محاسبه می‌گردد. همان طور که در شکل ۱ نشان داده شده است، روش پیشنهادی شامل سه گام اساسی می‌باشد:

گام ۱) مدل‌سازی اطلاعات جریانی حلقه: در ابتدا باید مسیرهای تکرار در بدنه حلقه تکرار شناسایی گردد. در هر تکرار حلقه، برای اجرا بر روی هر مسیر تکرار یک عبارت شرطی وجود دارد که به مقدار درست، ارزیابی و این عبارت برای هر مسیر ایجاد می‌گردد. هر عبارت شرط از متغیرهایی تشکیل شده که بر اساس مقدار آنها، عبارت شرطی ارزیابی می‌گردد. مقدار این متغیرها در نتیجه ارزیابی عبارت شرطی اثرگذار است که به آن متغیر کنترلی گفته می‌شود. بنابراین عبارت چگونگی تغییر این متغیرها بر روی هر مسیر نیز باید ایجاد گردد. با ایجاد این دو دسته عبارت نمادین، اطلاعات جریان داده‌ای و کنترلی حلقه مدل می‌گردد که در بخش ۳-۱ تشریح گردیده است.

گام ۲) تخمین تعداد تکرار هر مسیر تکرار: هر شرط تکرار بر اساس مقدار متغیرهای تشکیل‌دهنده آن به مقدار درست یا نادرست ارزیابی می‌گردد که سبب انتخاب مسیر مورد نظر برای تکرار خواهد شد. تعداد چندتایی‌هایی از مقدار این متغیرهای کنترلی را که سبب ارزیابی عبارت شرط برای یک مسیر تکرار به مقدار صحیح گردد، می‌توان یک تخمین برای بیشترین تعداد تکرار بر روی این مسیر در نظر گرفت. عبارت شرط تکرار می‌تواند یک چندوجهی بسته باشد و هر چندتایی از مقادیر متغیرهای کنترلی یک نقطه در این چندوجهی است که تعداد این نقاط درون چندوجهی تخمینی برای بیشترین تعداد تکرار بر روی این مسیر خواهد بود. اما تمامی این نقاط درون چندوجهی در زمان اجرا انتخاب نخواهند شد که این موضوع توسط قضیه ۳-۴ اثبات گردیده است. بنابراین حالاتی که در زمان اجرا رخ نخواهند داد، شناسایی و در محاسبه در نظر گرفته نمی‌شود. این موضوع با ایجاد تصویر چندوجهی و با استفاده از عبارت نشان‌دهنده چگونگی تغییر متغیرهای کنترلی انجام می‌گردد که در بخش ۳-۳ توضیح داده شده است. علاوه بر این برای کاهش بیش‌تخمین تعداد تکرار هر مسیر مقادیری از متغیرهای کنترلی که در چند مسیر مشترک بوده و در محاسبه تعداد تکرار لحاظ شده‌اند در محاسبات به حساب آورده شده‌اند تا میزان بیش‌تخمین کاهش یابد. این موضوع در بخش ۳-۵ تشریح گردیده است.

گام ۳) محاسبه بیشترین زمان اجرای حلقه: اجرای یک حلقه تکرار، ترکیبی (با تکرار) از مسیرهای تکرار خواهد بود که به یکی از مسیرهای پایان ختم می‌گردد. برای محاسبه بیشترین زمان اجرای حلقه، باید ترکیبی را یافت که سبب بیشترین زمان اجرا گردد. بدین منظور روش IPET [۱] توسعه یافته و با حل یک مسئله برنامه‌ریزی خطی صحیح این ترکیب محاسبه می‌گردد که در بخش ۳-۲ تشریح گردیده است.

عملکرد روش پیشنهادی با یک مثال در بخش ۳-۶ نشان داده شده است.

۳-۱ مدل‌سازی اطلاعات جریانی حلقه

در بدنه یک حلقه تکرار، مسیرهای مختلفی وجود دارد که در هر تکرار، حلقه از یکی از این مسیرها (مسیر تکرار) اجرا می‌گردد و در نهایت از یکی از این مسیرها (مسیر پایان) اجرای خود را خاتمه می‌دهد. در حقیقت دستورهای انشعاب شرطی در بدنه حلقه تکرار سبب ایجاد مسیرهای اجرایی متفاوت می‌گردد. انتخاب دستور بعد از دستورهای انشعاب شرطی وابسته به مقدار عبارت شرطی این دستورها خواهد بود. هر عبارت شرطی ترکیب فصلی یا عطفی عبارت رابطه‌ای است که عملوندهای یک عبارت رابطه‌ای می‌توانند متغیرها و ثوابت باشند. بنابراین مقدار عبارت شرطی یک دستور انشعاب شرطی، bc، به مقدار متغیرها، v، در نقطه‌ای از برنامه، w، که دستور به کار رفته، وابسته است و با $bc(v,w)$ آن را

می‌توان نمایش داد. بنابراین اگر حلقه بخواهد از یک مسیر خاص اجرا گردد باید متغیرها مقادیری داشته باشند که ترکیب عطفی عبارات شرطی بر روی این مسیر به مقدار درست ارزیابی شود. شرط تکرار از مسیر تکرار ip به صورت (۱) تعریف می‌گردد

$$ipCondition(ip, v, w) = \sum_{bc \in ip} bc(v, w) \qquad (۱)$$

برای حذف وابستگی عبارت شرطی به نقاط به کار گرفته شده در برنامه، متغیرهای v، با عبارات محاسباتی جایگزین می‌گردند که مقدار این متغیرها در w را بر اساس مقدار متغیرهایی (v_s) در ابتدای حلقه تکرار محاسبه می‌کند. بنابراین شرط مسیر تکرار به صورت (۲) خواهد بود

$$ipCondition(ip, v_s) = \sum_{bc \in ip} bc(v_s) \qquad (۲)$$

برای هر مسیر پایان tp در یک حلقه نیز می‌تواند شرط مسیر را به صورت (۳) تعریف کرد

$$tpCondition(tp, v_s) = \sum_{bc \in tp} bc(v_s) \qquad (۳)$$

این عبارات شرطی را می‌توان در قالب یک دستگاه نامعادلات خطی نیز نشان داد.

درست یا نادرست‌بودن این شرط‌ها وابسته به مقدار متغیرهای ظاهرشده در عبارات شرطی آن است. انتخاب هر یک از این مسیرها به مقدار متغیرهای کنترلی حلقه در ابتدای حلقه تکرار وابسته است که مقدار آنها از تکرار قبلی به دست آمده است. همچنین در تکرار بعدی مسیر تکرار بر اساس مقدار متغیرهای کنترلی در پایان این تکرار تعیین می‌گردد. بنابراین باید مشخص گردد که در هر مسیر تکرار، نحوه تغییر متغیرهای کنترلی حلقه به چه صورت است. مطابق (۴) عبارت نمادین نشان‌دهنده نحوه تغییر متغیرهای کنترلی حلقه تابعی خواهد بود که مقدار متغیرهای کنترلی را در انتهای مسیر تکرار فعلی cv_e (یا ابتدای مسیر تکرار بعدی) بر اساس مقدار متغیرهای کنترلی در ابتدای مسیر تکرار cv_s محاسبه می‌کند

$$cv_e = chgVal(ip, cv_s) \qquad (۴)$$

پس برای هر مسیر تکرار ip در بدنه حلقه تکرار یک شرط تکرار $ipCondition$ و یک عبارت نشان‌دهنده نحوه تغییر متغیرهای کنترلی $chgVal$ خواهیم داشت. اجرای یک حلقه تکرار، ترکیبی (با تکرار) از این مسیرهای تکرار خواهد بود که به یکی از مسیرهای پایان ختم می‌گردد. برای جزئیات استخراج و ایجاد عبارت نمادین به [۱۴] مراجعه گردد.

۳-۲ محاسبه بیشترین زمان اجرای حلقه

یک حلقه در هر تکرار از یک مسیر تکرار اجرا شده و در نهایت با اجرای روی یک مسیر پایان خاتمه می‌یابد. بنابراین زمان اجرای آن عبارت به صورت (۵) خواهد بود

$$T_{loop} = T_S + \sum_{i \in IP} (T_i \times X_i) + T_{tp} \qquad (۵)$$

که T_s زمان دستورهایی است که قبل از اولین تکرار اجرا می‌گردند مانند مقداردهی اولیه شمارنده و T_i زمان اجرای دنباله دستورها روی مسیر تکرار iام و T_{tp} زمان اجرای دنباله دستورها روی یکی از مسیرهای پایان است. X_i تعداد دفعات اجراشده مسیر تکرار iام و IP مجموعه کلیه مسیرهای تکرار است. در این مقاله فرض شده که T_s، T_i و T_{tp} از قبل توسط تحلیل سطح پایین [۱] محاسبه شده است. بنابراین (۶) را خواهیم داشت که T_{loop} مقدار بیشینه گردد

$$WCET_{loop} = WCET_s + \max\left(\sum_{i \in IP} WCET_i \times X_i\right) + \max_{t \in TP}(WCET_{tp}) \qquad (۶)$$

که $WCET_i$ بیشترین زمان اجرای دنباله‌ای از دستورها بر روی یک مسیر است که معلوم فرض شده است. باید در جمله دوم (۶)، X_i به نحوی انتخاب گردد که حاصل کل جمله دوم بیشینه گردد. این یک مسئله برنامه‌ریزی خطی صحیح است که باید مقدار X_iها به نحوی انتخاب گردد که تابع هزینه آن $\sum_{i \in IP} WCET_i \times X_i$ روی تعدادی محدودیت $X_i \le P_i$ بیشینه گردد. در بخش بعدی نحوه تخمین P_i شرح داده شده است.

۳-۳ تخمین بیشترین تکرار برای هر مسیر

در بیشتر حلقه‌های تکرار عبارات شرطی (bc) یک نامعادله خطی است. مجموعه جواب‌های یک نامعادله خطی برابر یک نیم‌فضا از فضای nبعدی است که توسط معادله خطی متناظر با نامعادله تعریف می‌گردد. n تعداد متغیرهای کنترلی حلقه است. یک شرط مسیر تکرار یا پایان یک دستگاه نامعادلات خطی از عبارات شرطی است. پاسخ این دستگاه نامعادلات خطی تقاطع (اشتراک) نیم‌فضاهای تعریف‌شده توسط هر یک از نامعادلات خطی مربوط به عبارت شرطی است. اگر هر یک از نیم‌فضاها محدب باشد پاسخ دستگاه که اشتراک این نیم‌فضاها است یک چندوجهی محدب خواهد بود. این چندوجهی می‌تواند بسته یا باز باشد.

کاهش یا افزایش متغیرهای کنترلی نیز در اغلب حلقه‌ها یک تابع خطی از متغیر کنترلی است. برای مثال برای متغیر کنترلی cv نحوه تغییر آن به صورت (۷) است

$$cv_{i+1} = a \times cv_i + b \qquad (۷)$$

که cv_i مقدار متغیر در تکرار iام، cv_{i+1} مقدار متغیر در تکرار $i+1$ام و a و b مقدار ثابت هستند. همچنین مقدار تغییر متغیر کنترلی به صورت (۸) تعریف می‌گردد

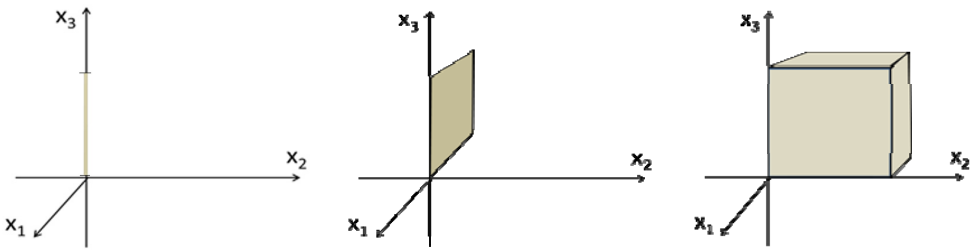
$$\Delta cv = cv_{i+1} - cv_i \qquad (۸)$$

برای حلقه‌هایی که شرط مسیر تکرار یک چندوجهی بسته است، هر نقطه درون این چندوجهی یک پاسخ برای دستگاه نامعادلات نشان‌دهنده شرط تکرار است. به عبارتی دیگر هر پاسخ دستگاه معادل چندتایی از مقدار متغیرهای کنترلی خواهد بود که عبارت شرط مسیر را درست ارزیابی می‌کند. بنابراین تعداد نقاط درون چندوجهی برابر تعداد تمام حالات مقداردهی متغیرهای کنترلی خواهد بود. پس با محاسبه تعداد این پاسخ‌ها می‌توان بیشترین تعداد که حلقه می‌تواند از این مسیر اجرا گردد را محاسبه نمود [۱۵] و [۱۶]. با توجه به نحوه تغییر متغیرهای کنترلی واضح است که بعضی از این نقاط هرگز انتخاب نخواهند شد و تخمین حاضر از بیشترین تکرار یک بیش‌تخمین[1] خواهد بود [۱۶].

برای این که مقدار تخمینی برای تعداد تکرار هر مسیر به مقدار واقعی آن نزدیک‌تر شود، تلاش می‌گردد که فضای nبعدی کاهش پیدا کند. به عبارت دیگر تصویری از چندوجهی در فضای mبعدی $(m < n)$ را به دست می‌آوریم مانند آنچه که در شکل ۲ نشان داده شده است. واضح است نقاط درون چندوجهی به دست آمده به مراتب کمتر از چندوجهی اصلی خواهد بود.

فرض می‌گردد که حلقه تکرار یک حلقه یکنواخت چندمسیری است.

1. Overestimation



(الف) (ب) (ج)

شکل ۲: چندوجهی و تصویر آن، (الف) چندوجهی، (ب) تصویر بر روی صفحه مختصات (x_1,x_2) و (ج) تصویر بر روی محور مختصات x_3.

یعنی در بدنه حلقه تکرار چندمسیر برای اجرا وجود دارد و نحوه تغییر متغیرهای کنترلی حلقه یکنواخت است. یک متغیر یکنواخت است اگر متغیر افزایشی یا کاهشی باشد:

یک متغیر v افزایشی است اگر در تمامی تکرارهای حلقه (iteration) خاصیت زیر را دارا باشد

$$\forall i \in iteration :: v_{i+1} \geq v_i$$

و یک متغیر کاهشی نامیده می‌شود اگر

$$\forall i \in iteration :: v_{i+1} \leq v_i$$

بنابراین برای تخمین تعداد تکرار هر مسیر مراحل زیر باید انجام گردد:

۱) با استفاده از یک حل‌کننده محدودیت [۱۷] شرط مسیر را تست کرده و اگر شرط به نادرست ارزیابی گردید، تعداد تکرار آنها صفر لحاظ می‌گردد. بدین ترتیب مسیرهای اجرانشدنی[1] شناسایی می‌گردد که در نزدیک‌بودن تخمین به مقدار واقعی بیشترین تعداد تکرار تأثیر بسزایی دارد [۱].

۲) اگر مقدار یک متغیر کنترلی در مسیر تغییری نداشت $(\Delta cv = ۰)$ آن گاه تصویر چندوجهی را بر روی سایر ابعاد به جز این بعد باید محاسبه نمود. در شکل ۲- ب مثالی نشان داده شده است.

۳) تصویر چندوجهی را برای هر بُعد به دست می‌آوریم. نتیجه حاصل یک پاره‌خط خواهد بود که بر اساس آن مقدار شروع V_s و مقدار پایانی V_e متغیر کنترلی متناظر بُعد به دست می‌آید که نمونه‌ای در شکل ۲- ج نشان داده شده است.

۴) بیشترین تعداد تکرار بر روی این مسیر (مطابق با قضیه بخش ۴-۱) برابر با بیشترین تعداد مقادیری است که می‌تواند به یکی از متغیرهای کنترلی انتساب داد یعنی بر اساس نتایج گام قبلی برابر خواهد بود با $V_e - V_s + ۱$. اگر نحوه تغییر متغیر کنترلی خطی به صورت (۷) باشد (فرض کنید که تغییر متغیر افزایشی است) تعداد مقادیر ممکن برای انتساب به متغیر cv و یا بیشترین دفعات اجرای این مسیر بر اساس این متغیر f_{cv} به صورت (۹) محاسبه می‌گردد [۱۸]

$$f_{cv} = \begin{cases} \left\lfloor \log_a \left(۱ + \frac{(V_e - V_s)\times(a-۱)}{b+(a-۱)\times V_s}\right) \right\rfloor + ۱ \\ \quad \text{if } a > ۱ \text{ and } V_e > V_s \\ \left\lfloor \frac{V_e - V_s}{b} \right\rfloor + ۱ \text{ if } a = ۱ \text{ and } V_e > V_s \end{cases} \quad (۹)$$

اکنون برای مسیر تکرار ip و متغیرهای کنترلی وابسته به آن، CV، می‌توان تعداد تکرار را از (۱۰) به دست آورد

$$P_{ip} = \min_{cv \in CV} f_{cv} \quad (۱۰)$$

حال محدودیت $X_{ip} < P_{ip}$ را به محدودیت‌های مسئله برنامه‌ریزی خطی اضافه کرده و مراحل فوق برای سایر مسیرهای تکرار دوباره انجام می‌شود.

۳-۴ قضیه

اگر تمامی متغیرهای کنترلی حلقه یکنواخت باشند بر اساس قضیه زیر بیشترین دفعات اجرای یک مسیر برابر خواهد بود با تعداد مقادیر ممکن یکی از متغیرهای کنترلی حلقه مرتبط با آن مسیر. قضیه زیر نشان می‌دهد که یک مقدار مشخص برای یک متغیر در تکرارهای مختلف بر روی آن مسیر دوباره ظاهر نمی‌گردد (با توجه به مرحله دوم که صورت گرفته است).

قضیه: برای یک حلقه با متغیرهای کنترلی یکنواخت، اگر در یک اجرا بر روی یک مسیر خاص، یک چندتایی از مقدار متغیرهای کنترلی $(cv_1, cv_2, cv_3, \ldots, cv_k)$ وجود داشته باشد، هیچ چندتایی دیگر $(cv'_1, cv'_2, cv'_3, \ldots, cv'_k)$ نمی‌توان یافت که سبب اجرای حلقه بر روی این مسیر گردد و داشته باشیم $cv_i = cv'_i$ و برای $i \neq j$ و $j : ۱, \ldots, k$ داشته باشیم $cv_j \neq cv'_j$.

اثبات: فرض نمایید که $(cv_1, cv_2, cv_3, \ldots, cv_k)$ یک چندتایی از مقدار متغیرهای کنترلی است که حلقه بر روی مسیر ip اجرا خواهد شد. بعد از اجرا بر روی مسیر ip و بر اساس $ChgVal_{ip}$ چندتایی بعدی به صورت $(cv'_1, cv'_2, cv'_3, \ldots, cv'_k)$ خواهد بود. با توجه به این که همه متغیرهای حلقه یکنواخت هستند و متغیرهایی که مقدارشان تغییر نمی‌کند حذف گردیده‌اند خواهیم داشت (فرض کنید متغیرها یکنواخت افزایشی باشند): $cv_1 < cv'_1$، $cv_2 < cv'_2$، ... و $cv_k < cv'_k$. با ارزیابی شرط مسیر $ipCondition$ مربوط به ip با چندتایی $(cv'_1, cv'_2, cv'_3, \ldots, cv'_k)$ دو حالت ممکن است رخ بدهد:

- $ipCondition((cv'_1, cv'_2, cv'_3, \ldots, cv'_k))$ به $true$ ارزیابی می‌گردد و این بدین معنی است که تکرار بعدی بر روی این مسیر خواهد بود. بنابراین $cv_i = cv'_i$ در تکرار بعدی رخ نخواهد داد.
- $ipCondition((cv'_1, cv'_2, cv'_3, \ldots, cv'_k))$ به $false$ ارزیابی می‌گردد و این بدین معنی است که تکرار بعدی بر روی یک مسیر دیگر به جز ip خواهد بود. حتی اگر در مسیرهای تکرار بعدی مقدار cv'_i هیچ گونه تغییری نداشته باشد و مسیر ip با چندتایی $(cv''_1, cv''_2, cv''_3, \ldots, cv''_k)$ که $cv''_i = cv'_i$ دوباره برای اجرا انتخاب گردد، باز هم $cv_i < cv'_i$ برقرار خواهد بود و $cv_i = cv'_i$ رخ نخواهد داد.

۳-۵ کاهش بیش‌تخمین

اگر مجموعه مقادیر یک متغیر کنترلی خاص برای چند مسیر تکرار حلقه یکسان باشد، تکرار بر روی یک مسیر از تعداد تکرار بر روی

1. Infeasible Path

```
A       i=1;
B       j=11;
C       while (i<10) {
D         if (i<j) {
E           i=i+1;
F           j=j-2;
G         }
H         else
K           i=i+1;
L         if (i>5)
M           i=i+1;
N       }
P
```

شکل ۳: قطعه کد برای تخمین بیشترین زمان اجرای حلقه.

مسیرهای دیگر خواهد کاست. بنابراین مجموع تعداد تکرار بر روی این مسیرها باید محدود گردد به بیشترین تعداد مقادیر ممکن که به این متغیر می‌توان انتساب داد. به عبارت دیگر تصویر دو یا بیشتر چندوجهی مرتبط به دو یا چند مسیر تکرار بر روی یک بُعد یا متغیر کنترلی با هم برابر می‌گردند

$$Proj_{cv}(Q_1) = Proj_{cv}(Q_2) = \ldots = Proj_{cv}(Q_m) \qquad (11)$$

Q_k چندوجهی متناظر با kامین مسیر تکرار ip_k است. با توجه به این که متغیرهای کنترلی یکنواخت می‌باشند محدودیت زیر را می‌توان به مسئله برنامه‌ریزی خطی صحیح اضافه نمود

$$X_{ip_1} + X_{ip_2} + \ldots + X_{ip_m} \le \max(f^{cv}_{ip_1}, f^{cv}_{ip_2}, \ldots, f^{cv}_{ip_m}) \qquad (12)$$

$f^{cv}_{ip_i}$ بیشترین تعداد تکرار مسیر ip_i بر اساس متغیر cv است (بیشترین تعداد مقادیری که به این متغیر می‌توان انتساب داد).

همچنین تصویر دو یا بیشتر چندوجهی مرتبط به دو یا چند مسیر تکرار بر روی یک بُعد (متغیر کنترلی) مانند (۱۱) ممکن است به طور دقیق با هم برابر نبوده اما مطابق با (۱۳) با هم اشتراک داشته باشند

$$Proj_{cv}(Q_1) \cap Proj_{cv}(Q_2) \cap \ldots \cap Proj_{cv}(Q_m) \neq \varnothing \qquad (13)$$

برای هر دو چندوجهی i و j که تصویر آنها بر روی یک بُعد (متغیر کنترلی cv) اشتراک داشته باشد (۱۴) را خواهیم داشت

$$Proj_{cv}(Q_i) \cap Proj_{cv}(Q_j) = LS_{i,j} \qquad (14)$$

$LS_{i,j}$ نشان‌دهنده پاره‌خطی بر روی بُعد متناظر متغیر کنترلی cv است. مطابق با آنچه در بخش چهارم بیان شد بیشترین تعداد مقادیر که می‌توان به متغیر کنترلی متناظر آن نسبت داد بر اساس (۹) و پاره‌خط $LS_{i,j}$ قابل محاسبه و به ترتیب برای مسیر i و j برابر با $f^{cv}_{ls_i}$ و $f^{cv}_{ls_j}$ خواهد بود. اگر تعداد تکرار مسیرهای ip_i و ip_j بر اساس متغیر کنترلی cv به ترتیب $f^{cv}_{ip_i}$ و $f^{cv}_{ip_j}$ و تعداد تکرار این مسیرها بر اساس پاره‌خط باقیمانده (به جز $LS_{i,j}$) از پاره‌خط متناظر با $Proj_{cv}(Q_i)$ و $Proj_{cv}(Q_j)$ به ترتیب $f^{cv}_{ip_i/ls}$ و $f^{cv}_{ip_j/ls}$ باشد (۱۵) را خواهیم داشت

$$\begin{aligned} f^{cv}_{ip_i} &= f^{cv}_{ip_i/ls} + f^{cv}_{ls_i} \\ f^{cv}_{ip_j} &= f^{cv}_{ip_j/ls} + f^{cv}_{ls_j} \end{aligned} \qquad (15)$$

بر اساس (۱۰) برای مسیر i رابطه‌های $P_{ip_i} \le f^{cv}_{ip_i}$ و $P_{ip_j} \le f^{cv}_{ip_j}$ برقرار است و با توجه به (۱۵) می‌توانیم محدودیت (۱۶) را به مسئله برنامه‌ریزی خطی صحیح اضافه نماییم

$$X_{ip_i} + X_{ip_j} \le f^{cv}_{ip_i/ls} + f^{cv}_{ip_j/ls} + \max(f^{cv}_{ls_i}, f^{cv}_{ls_j}) \qquad (16)$$

برای هر دو یا چند مسیر که (۱۳) برای آنها برقرار است می‌توان محدودیتی مانند (۱۶) را به مسئله برنامه‌ریزی خطی صحیح اضافه نمود. افزودن محدودیت‌های بیشتر مانند (۱۲) و (۱۶) به مسئله برنامه‌ریزی خطی صحیح سبب خواهد گردید که مقدار تخمینی دقیق‌تر باشد (به مقدار واقعی نزدیک‌تر است).

۶-۳ مثال

در شکل ۳ یک قطعه کد دارای یک حلقه while نشان داده شده است.

در بدنه حلقه تکرار چهار مسیر تکرار وجود دارد

$$\begin{aligned} ip_1 &= C\ D\ E\ F\ G\ L\ M\ N \\ ip_2 &= C\ D\ E\ F\ G\ L\ N \\ ip_3 &= C\ D\ H\ K\ L\ M\ N \\ ip_4 &= C\ D\ H\ K\ L\ N \end{aligned}$$

همچنین یک مسیر پایان $tp = CK$ وجود دارد.

شرط تکرار و خاتمه برای هر یک از مسیرهای فوق در قالب یک دستگاه نامعادلات خطی و نحوه تغییر شمارنده در هر مسیر به صورت زیر است

$$ipCondition(ip_1) = \begin{cases} i \ge 1 \\ i < 10 \\ i < j \\ i + 1 > 5 \\ j \le 11 \end{cases} \quad chgVal(ip_1): i = i + 2, j = j - 2$$

$$ipCondition(ip_2) = \begin{cases} i \ge 1 \\ i < 10 \\ i < j \\ i + 1 \le 5 \\ j \le 11 \end{cases} \quad chgVal(ip_2): i = i + 1, j = j - 2$$

$$ipCondition(ip_3) = \begin{cases} i \ge 1 \\ i < 10 \\ i \ge j \\ i + 1 > 5 \\ j \le 11 \end{cases} \quad chgVal(ip_3): i = i + 2$$

$$ipCondition(ip_4) = \begin{cases} i \ge 1 \\ i < 10 \\ i \ge j \\ i + 1 \le 5 \\ j \le 11 \end{cases} \quad chgVal(ip_4): i = i + 1$$

در عبارات فوق آیتم $i \ge 1$ و $j \le 11$ بر اساس مقدار اولیه‌شان و همچنین افزایش/ کاهش یکنواخت در بدنه حلقه به هر شرط مسیر اضافه گردیده است. نکته دیگر که باید مورد توجه قرار گیرد عبارت شرط $i < 5$ بر اساس مقدار i در دستور L است. بنابراین این شرط به $i+1<5$ تغییر پیدا نموده که بر اساس مقدار i در ابتدای حلقه تکرار خواهد بود.

مسیر تکرار ip_1 یک مسیر اجرانشدنی نیست و همچنین هر دو متغیر کنترلی روی این مسیر تغییر می‌کند. بنابراین تصویر چندوجهی بر محور i و j به ترتیب به صورت $i > 4 \text{ and } i < 10$ و $j > 5 \text{ and } j \le 11$ است که بیشترین مقدار ممکن برای i مقدار ۹ و کمترین مقدار ۵ خواهد بود. همچنین برای j بیشترین مقدار ۱۱ و کمترین مقدار ۶ خواهد

جدول ۱: تعداد تکرار محاسبه‌شده برای مثال شکل ۳.

ردیف	نام مسیر	P_{IP}	$P_{IP,CV}$
۱	IP_1	۲۰	۳
۲	IP_2	۳۴	۴
۳	IP_3	∞	۳
۴	IP_4	∞	۴

خواهد بود. با توجه به (۹) و (۱۰) می‌توان تعداد تکرار از این مسیر را به صورت (۱۷) محاسبه نمود

$$Pip_1 = \max\left(\left\lfloor \frac{9-5}{2} \right\rfloor + 1, \left\lfloor \frac{11-6}{2} \right\rfloor + 1\right) = 3 \quad (17)$$

به همین صورت برای مسیر تکرار ip_2 (۱۸) را داریم

$$Pip_2 = \max\left(\left\lfloor \frac{4-1}{1} \right\rfloor + 1, \left\lfloor \frac{11-5}{2} \right\rfloor + 1\right) = 4 \quad (18)$$

در دو مسیر تکرار بعدی مقدار j تغییر نمی‌کند. بنابراین باید تصویر چندوجهی نشان‌دهنده دستگاه نامعادلات خطی آنها را بر روی محور i به دست آورد و سپس تعداد تکرار را بر اساس (۹) و (۱۰) به صورت (۱۹) محاسبه نمود

$$\begin{aligned} Pip_3 &= \left\lfloor \frac{9-5}{2} \right\rfloor + 1 = 3 \\ Pip_4 &= \left\lfloor \frac{4-1}{1} \right\rfloor + 1 = 4 \end{aligned} \quad (19)$$

در جدول ۱ تعداد تکرار محاسبه‌شده توسط روش مبتنی بر تطبیق الگو در این مقاله ($P_{ip,cv}$) و تعداد کل مقادیر انتسابی ممکن به متغیرهای کنترلی (تمامی نقاط درون چندوجهی) (P_{ip}) محاسبه‌شده با روش [۱۶] آمده است. همان طور که مشاهده می‌گردد برای مسیرهای ip_3 و ip_4 توسط روش [۱۶] مقدار بی‌نهایت بوده که به دلیل بازبودن چندوجهی است اما در روش ارائه‌شده در این مقاله این محدودیت اثرگذار نبوده و همچنین روش ارائه‌شده در این مقاله بیش‌تخمین کمتری را دارا است.

با فرض این که زمان اجرای هر دستور در قطعه کد فوق ۱۰ واحد زمانی است خواهیم داشت

$$WCET_s = 20 \,,\, WCET_{tp} = 10 \,,\, WCET_{ip_1} = 60$$
$$WCET_{ip_2} = 50 \,,\, WCET_{ip_3} = 50 \,,\, WCET_{ip_4} = 40$$

با توجه به (۶) باید مقدار تابع هزینه (۲۰) بیشینه گردد

$$WCET_{loop} = 20 + \max(60 \times X_{ip_1} + 50 \times X_{ip_2} + 50 \times X_{ip_3} + 40 \times X_{ip_4}) + 10 \quad (20)$$

محدودیت‌ها عبارتند از ($X_{ip} \le P_{ip}$)

$$Xip_1 \le 3 \,,\, Xip_2 \le 4 \,,\, Xip_3 \le 3 \,,\, Xip_4 \le 4 \quad (21)$$

با حل مسئله برنامه‌ریزی خطی صحیح بر اساس (۲۰) و (۲۱) $WCET_{loop} = 720$ به دست می‌آید.

این مقدار تخمینی به طور تقریبی ۱۰۰ درصد بیش‌تخمین نسبت به بیشترین زمان اجرای واقعی را دارد. در ادامه سعی می‌گردد بر اساس آنچه در این مقاله به عنوان نوآوری مطرح شده است این بیش‌تخمین نیز کاهش یابد.

بر اساس آنچه در بخش ۳-۵ تشریح گردید، تصویر چندوجهی مربوط به مسیرهای ip_1 و ip_2 بر روی بعد متناظر با متغیر i و تصویر چندوجهی مربوط به مسیرهای ip_2 و ip_4 بر روی بعد متناظر با متغیر j یکسان شده‌اند. بنابراین طبق (۱۱) و (۱۲) ۲ محدودیت $X_{ip_2} + X_{ip_4} \le 4$ و $X_{ip_1} + X_{ip_2} \le 3$ را می‌توان به مسئله برنامه‌ریزی خطی صحیح با تابع هزینه (۲۰) اضافه نمود. با حل مسئله برنامه‌ریزی خطی صحیح بیشترین زمان اجرا $WCET_{loop} = 410$ به دست می‌آید که ۵۷٪ بهبود نسبت به قبل داشته [۵] و فقط حدود ۱۰٪ بیش‌تخمین نسبت به مقدار واقعی بیشترین زمان اجرا دارد.

جدول ۲: قطعات کد نمونه برای ارزیابی روش ارائه‌شده.

نام قطعه کد	قطعه کد برنامه	نام برنامه محک و منبع

محک ۱	`for (i = 0; i < 10; i++) {`
	`  a = IN;`
	`  if (i < 5) {`
	`  a = a & 0x0F;`
	`  OUT = num_to_lcd (a);`
	`  }`
	`}`
		lcdnum در [۱۱]

محک ۲	`for (j = 0; j < 100; j++) {`
	`  if (a[k][j] == 1) j++;`
	`  else a[k][j] = -1;`
	`}`
		شکل ۱ در [۱۰]

محک ۳	`while (k < j) {`
	`  j* = k;`
	`  k/ = 2;`
	`}`
		fft۱ در [۱۱]

محک ۴	`for (i = 1; i <= 100; i++) {`
	`  a = -i*(nm1 + i);`
	`  b+= 2;`
	`  d = 10*(a*d + b);`
	`  c = b + a/c;`
	`  del = c*d;`
	`  h* = del;`
	`  if (del < 10000) {`
	`  ans = h*-x;`
	`  return ans;`
	`  }`
	`}`
		expint در [۱۱]

محک ۵	`b=5;`
	`for (i = 1; i <= 100; i++) {`
	`  b+= 2;`
	`  c = 5*i + b;`
	`  if (c > 42)`
	`  break;`
	`}`
		محک ارائه‌شده در این مقاله

٤- ارزیابی

برای ارزیابی، حلقه‌هایی از برنامه‌های محک Malardalen [۱۹] و برنامه‌های محک موجود در گزارش‌های سایر پژوهش‌ها انتخاب گردید تا توانایی روش ارائه‌شده در این مقاله نشان داده شود. این قطعات کد در جدول ۲ نشان داده شده‌اند. در این بخش به طور مشخص روش ارائه‌شده با روش [۱۲] مقایسه گردیده است به دلیل این که این روش جزء پژوهش‌های اخیر بوده و نسبت به سایر روش‌ها محدودیت کمتری داشته و از دقت بالاتری برخوردار است. البته این روش پیاده‌سازی نشده و نتایج ارائه‌شده برای روش [۱۲] به صورت دستی به دست آمده است. همچنین فرض شده که زمان اجرای همه دستورها در بدنه حلقه ثابت و معلوم است و بیشترین زمان اجرای دستورها قبل از شروع تکرار ($WCET_S$) و همچنین بیشترین زمان اجرای دستورها بر روی مسیر خاتمه ($WCET_{tp}$) در (۶) قابل ملاحظه نبوده و در محاسبات نادیده گرفته شده است.

محک ۱، یک حلقه است که در بدنه آن یک دستور شرطی بر اساس مقدار متغیر کنترلی وجود دارد. اما متغیر حلقه در بدنه این دستور شرط تغییری ندارد و بنابراین روش ارائه‌شده در [۱۲] این دستور شرط را نادیده گرفته و فرض می‌نماید که حلقه فقط یک مسیر برای اجرا دارد. در پایان این روش تعداد تکرار را برای این حلقه ۱۰ محاسبه می‌نماید.

اما روش ارائه‌شده در این مقاله، دو مسیر تکرار در بدنه حلقه در نظر می‌گیرد که توسط دستور شرط موجود در بدنه ایجاد شده است. شرط تکرار برای هر یک از این مسیرها به صورت (۲۲) خواهد بود

$$\begin{aligned} &ipCond_{ip_1} : i \geq 0 \text{ and } i < 10 \text{ and } i < 5 \\ &ipCond_{ip_2} : i \geq 0 \text{ and } i < 10 \text{ and } i \geq 5 \end{aligned} \quad (22)$$

تعداد تکرار برای هر مسیر تکرار بر اساس (۹) برابر پنج خواهد بود، در صورتی که بیشترین زمان اجرا برای هر مسیر تکرار $WCET_{ip_1} = t_1$, $WCET_{ip_2} = t_2$ and $t_1 > t_2$ باشد، بیشترین زمان اجرا به صورت (۲۳) خواهد بود

$$WCET_{loopbody} = \max(WCET_{ip_1}, WCET_{ip_2}) = t_1 \quad (23)$$

در نهایت بیشترین زمان اجرای حلقه $WCET_{loop}$ محاسبه‌شده توسط روش [۱۲] و روش ارائه‌شده در این مقاله به ترتیب $10\times t_1$ و $5\times t_1 + 5\times t_2$ خواهد بود که روش ارائه‌شده در این مقاله دقیق‌تر است. در عمل روش ارائه‌شده در این مقاله نسبت به [۱۲] دقیق‌تر تخمین زده می‌شود چون در [۱۲] دستور شرط در اجرای حلقه نادیده گرفته می‌شود و تعداد تکرار دستورها در بدنه شرط را برابر با تعداد تکرار حلقه در نظر می‌گیرد.

در محک ۲ یک انشعاب شرطی در بدنه حلقه با شرط نامعین $a[k][j] == 1$ وجود دارد. مراجع [۱۲] و [۵] تعداد تکرار را با بیش‌تخمین به ۱۰۰ تخمین می‌زنند. روش ارائه‌شده در این مقاله می‌تواند برای این حلقه بیشترین زمان اجرا را تخمین بزند. برای این شرط مسیر تکرار (۲۴) خواهد بود

$$\begin{aligned} &ipCond_{ip_1} : j \geq 0 \text{ and } j < 100 \text{ and } a[k][j] == 1 \\ &ipCond_{ip_2} : j \geq 0 \text{ and } j < 100 \text{ and } a[k][j] \neq 1 \end{aligned} \quad (24)$$

هر عبارت شرطی در انشعاب که به هر دلیلی نامعین بوده و یا نتواند محدودیت‌های روش ارائه‌شده را برآورده نماید می‌تواند از شرط تکرار حذف گردد. بنابراین (۲۴) به (۲۵) تبدیل خواهد شد

$$\begin{aligned} &ipCond_{ip_1} : j \geq 0 \text{ and } j < 100 \\ &ipCond_{ip_2} : j \geq 0 \text{ and } j < 100 \end{aligned} \quad (25)$$

پس شرط مسیر تکرار برای هر دو مسیر یکسان گردیده است. نحوه تغییر متغیر کنترلی j در دو مسیر تکرار به ترتیب $chgVal(ip_1) : j = j + 2$ و $chgVal(ip_2) : j = j + 1$ است.

با توجه به (۹) تعداد تکرار برای ip_1 برابر ۵۰ و برای ip_2 برابر ۱۰۰ خواهد بود. بنابراین مسئله برنامه‌ریزی خطی برای این محک در صورتی که $WCET_{ip_1} = t_1$، $WCET_{ip_2} = t_2$ و $t_1 < t_2$ به صورت (۲۶) است

$$\begin{aligned} &WCET_{loop} = \max\{t_1 \times X_1 + t_2 \times X_2\} , \\ &X_1 \leq 50 , X_2 \leq 100 \end{aligned} \quad (26)$$

چون شرط تکرار هر دو مسیر تکرار یکسان شده است با توجه به (۱۲) محدودیت $X_1 + X_2 \leq \max(50,100)$ به مسئله برنامه‌ریزی خطی اضافه می‌گردد.

برای مقادیر مختلف t_1 و t_2، تخمین $WCET_{loop}$ در روش ارائه‌شده در این مقاله کمتر یا مساوی تخمین ارائه‌شده در روش [۱۲] یعنی $100\times t_1$ خواهد بود که نشان از دقت بالاتر روش ارائه‌شده در این مقاله دارد.

محک ۳ قابل تحلیل توسط روش [۱۲] نیست. این مثال توسط روش ارائه‌شده در این مقاله نیز قابل تحلیل نیست چون تصویر چندوجهی یک پاره‌خط نخواهد بود. اگر عبارت $k \geq 0$ را بتوان به شرط تکرار اضافه نمود روش ارائه‌شده در این مقاله می‌تواند بیشترین زمان اجرا برای این حلقه را تخمین بزند. این شرط می‌تواند از کد برنامه قبل از حلقه استخراج گردد.

محک ۴ یک حلقه است که اجرای آن با یک انشعاب شرطی در بدنه حلقه خاتمه می‌پذیرد و به عبارت دیگر، یک حلقه چندمسیری با خاتمه ناگهانی[1] است. روش [۱۲] انشعاب شرطی را نادیده گرفته و تعداد تکرار را برای این حلقه ۱۰۰ تکرار تخمین می‌زند. روش ارائه‌شده در این مقاله همین تخمین را ارائه می‌نماید. چون عبارت شرطی غیر خطی بوده از عبارت شرط تکرار مسیر حذف می‌گردد و تعداد تکرار ۱۰۰ تخمین زده خواهد شد. اما اگر مانند محک ۵، عبارت شرط خطی می‌بود روش [۱۲] دوباره همان مقدار قبلی را تخمین می‌زد و بیشترین زمان اجرای حلقه را $100\times t$ ارائه می‌نمود که t بیشترین زمان اجرای بدنه حلقه است. اما روش ارائه‌شده در این مقاله شرط مسیر تکرار را به صورت (۲۷) ایجاد می‌نماید

$$i \geq 1 \text{ and } b \geq 5 \text{ and } i \leq 100 \text{ and } 5\times i + b + 2 \leq 42 \quad (27)$$

تعداد تکرار محاسبه‌شده برای این مسیر برابر ۷ خواهد بود و بیشترین زمان اجرای حلقه $7\times t$ محاسبه شده که بسیار دقیق‌تر است. در محک ۵ نشان داده شده که اگر یک حلقه چندمسیری با بیش از یک متغیر کنترلی حلقه داشته باشیم در روش ارائه‌شده در این مقاله قابل شناسایی بوده و اثرات آن بر تعداد تکرار حلقه لحاظ می‌گردد.

۹۲٪ از مجموع ۱۱۴ حلقه تکرار در برنامه‌های محک Malardalen قابل تحلیل هستند [۱۴]. به عبارت دیگر می‌توان برای آنها عبارت نمادین ذکرشده را ایجاد نمود و با روش ارائه‌شده بیشترین زمان اجرای آنها را تخمین زد. بیش از ۱۶٪ این حلقه‌های تکرار چندمسیری هستند و در مواردی با خاتمه ناگهانی که روش ارائه‌شده در این مقاله تخمین دقیق‌تری برای این نوع از حلقه‌ها ارائه می‌نماید. تغییر متغیرهای کنترلی حلقه در تمامی حلقه‌های تکرار یکنواخت بودند. فقط دو حلقه تکرار، کمتر از ۲٪، تغییر متغیر به صورت غیر خطی بودند که امکان تخمین وجود نداشت. بنابراین فرض خطی و یکنواخت‌بودن تغییر متغیرهای کنترلی در این روش محدودکننده نیست و روش ارائه‌شده در این مقاله کاربردی است.

۵- نتیجه‌گیری و کارهای آینده

تخمین بیشترین زمان اجرای حلقه‌های تکرار یکی از مهم‌ترین مسایل تحلیل بیشترین زمان اجرای وظیفه‌ها در نرم‌افزارهای بی‌درنگ است. در روش‌های تطبیق الگو برای محاسبه تعداد تکرار، ساختار و مکان شرط‌های تست‌کننده شمارنده و مکان و چگونگی تغییر شمارنده در بدنه حلقه از مسایلی هستند که سبب عدم تطبیق الگو و در نتیجه شکست محاسبه تعداد تکرار می‌گردد. یک روش برای حل این مشکل که کمترین وابستگی را به شرط‌ها و دستورهای تغییردهنده شمارنده در بدنه حلقه تکرار داشته باشد از قبل ارائه گردیده است. در این روش تخمین تعداد تکرار بر اساس

1. Abrupt Termination

تعداد مقادیری که می‌توان در طی اجرا و تکرار حلقه به متغیرهای کنترلی یا شمارنده‌ها انتساب داد محاسبه می‌گردد. البته در نظر گرفتن همه مقادیر منجر به بیش‌تخمین می‌گردد. این بیش‌تخمین با در نظر نگرفتن مقادیری از متغیرها که در حین اجرای حلقه تکرار رخ نخواهند داد، کاهش و تخمین بیشترین زمان اجرای حلقه تکرار بهبود می‌یابد. در این مقاله بر اساس مقدار متغیرهای کنترلی که در مسیرهای مختلف مقادیر یکسانی دارند، وابستگی بین مسیرها شناسایی گردید. این وابستگی‌ها در قالب محدودیت‌هایی به مسئله برنامه‌ریزی خطی صحیح اضافه شد و دقت روش ارائه‌شده افزایش یافت. روش ارائه‌شده گرچه محدود به حلقه‌ها با متغیرهای یکنواخت و تغییر خطی بود اما مطالعه برنامه‌های محک موجود نشان داد که این روش کاربردی است و نتایج قابل قبولی ارائه می‌نماید.

در کارهای آینده تلاش می‌گردد که برای چندوجهی باز نشان‌دهنده شرط مسیر مانند محک ۳ و همچنین تغییر غیر یکنواخت شمارنده برای یک مسیر از چند مسیر یکنواخت، روش ارائه‌شده توسعه یابد.

مراجع

[1] R. Wilhelm, *et al*., "The worst-case execution time problem-overview of approaches and survey of tools," *Trans. on Embedded Computing Sys.*, vol. 7, no. 3, Article No. 36, Apr. 2008.

[2] R. Chapman, *Static Timing Analysis and Program Proof*, Ph.D. Dissertation, Univ. of York, 1995.

[3] J. Gustafsson, A. Ermedahl, C. Sandberg, and B. Lisper, "Automatic derivation of loop bounds and infeasible paths for WCET analysis using abstract execution," in *Proc. 27th IEEE Real-Time Systems Symp.*, *RTSS'06,* pp. 57-66, Rio de Janeiro, Brazil, 5-8 Dec. 2006.

[4] M. Michiel, A. Bonenfant, H. Casse, and P. Sainrat, "Static loop bound analysis of C programs based on flow analysis and abstract interpretation," in *Proc. IEEE Int. Conf. on Embedded and Real-Time Computing Systems and Applications, RTCSA'08*, pp. 161-166, Kaohsiung, Taiwan, 25-27 Aug. 2008.

[۵] پ. سخائی‌نیا، "روشی مبتنی بر تطبیق الگو برای تخمین بیشترین زمان اجرای حلقه‌ها،" *هشتمین کنفرانس بین‌المللی فناوری اطلاعات و دانش*، صص. ۷۶۶-۷۶۰، همدان، شهریور ۱۳۹۵.

[6] C. Healy, M. Sjodin, V. Rustagi, D. Whalley, and R. Engelen, "Supporting timing analysis by automatic bounding of loop iterations," *J. of Real-Time Systems*, vol. 18, no. 2-3, pp. 129-156, May 2000.

[7] Tidorum Bound-T tool homepage. www.tidorum.fi/bound-t, 2008.

[8] N. Holsti, L. Thomas, and S. Saarinen, "Worst-case execution-time analysis for digital signal processors," in *Proc. of 10th European Signal Processing Conf., EUSIPCO'00*, 14 pp., Tampere, Finland, 4-8 Sept. 2000.

[9] A. Prantl, M. Schordan, and J. Knoop, "TuBound-a conceptually new tool for worst-case execution time analysis," in *Proc. 8th Int. Workshop on Worst-Case Execution Time Analysis, WCET'08*, volpp. 141-148, Prague, Czech Republic, 2008.

[10] -, *aiT Tool: The Industry Standard for Static Timing Analysis*, 2007. http://www.absint.com/ait/

[11] C. Cullmann and F. Martin, "Data-flow based detection of loop bounds," in *Proc. 7th Int. Workshop on Worst-Case Execution Time Analysis, WCET'07*, 6 pp., Italy, Jul. 2007.

[12] J. Knoop, L. Kovacs, and J. Zwirchmayr, "Symbolic loop bound computation for WCET analysis, in *Perspectives of Systems Informatics*, E. Clarke, I. Virbitskaite, and A. Voronkov, Eds. Springer Berlin Heidelberg, pp. 227-242, 2012.

[13] A. Biere, J. Knoop, L. Kovacs, and J. Zwirchmayr, "The auspicious couple: symbolic execution and WCET analysis," in *Proc. of the 13th International Workshop on Worst-Case Execution Time Analysis*, vol. 30, pp. 53-63, 9-9 Jul. 2013.

[14] S. Parsa and M. Sakhaei-Nia, "Modeling flow information of loops using compositional condition of controls," *The J. of Supercomputing*, vol. 71, no. 2, pp. 508-536, Feb. 2015.

[15] S. Bygde, A. Ermedahl, and B. Lisper, "An efficient algorithm for parametric WCET calculation," *J. of Systems Architecture*, vol. 57, no. 6, pp. 614-624, Jun. 2011.

[16] S. Parsa and M. Sakhaei-Nia, "PLEA: parametric loop bound estimation in WCET analysis," *Turkish J. of Electrical Engineering & Computer Sciences*, vol. 24, no. 4, pp. 2135-2149, Apr. 2016.

[17] L. N. D. Moura and N. Bjorner, "Z3: an efficient SMT solver," *TACAS*, vol. 4963, pp. 337-340, Springer, 2008.

[18] D. Kebbal and P. Sainrat, "Combining symbolic execution and path enumeration in worst-case execution time analysis," in *Proc. 6th Int. Workshop on Worst-Case Execution Time Analysis, WCET'06*, vol. 4, 6 pp., Jul. 2006.

[19] J. Gustafsson, A. Betts, A. Ermedahl, and B. Lisper, "The malardalen WCET benchmarks: past, present and future," in *Proc. 10th Int. Workshop on Worst-Case Execution Time Analysis, WCET'10,* vol. 15, pp. 136-146, 2010.

مهدی سخائی‌نیا تحصیلات خود را در مقاطع کارشناسی در سال‌های ۱۳۷۵ از دانشگاه فردوسی مشهد و مقاطع کارشناسی ارشد و دکتری به ترتیب در سال‌های ۱۳۸۵ و ۱۳۹۴ از دانشگاه علم و صنعت ایران در رشته مهندسی کامپیوتر گرایش نرم‌افزار دریافت نمود. ایشان هم اکنون استادیار گروه مهندسی کامپیوتر در دانشکده مهندسی دانشگاه بوعلی سینا است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: تحلیل بیشترین زمان اجرای برنامه‌ها، تحلیل ایستای برنامه‌ها، شناسایی آسیب‌پذیری برنامه‌ها، معماری سرویس‌گرا و مهندسی نرم‌افزار.

سعید پارسا در سال ۱۳۵۹ مدرک کارشناسی علوم کامپیوتر خود را از دانشگاه صنعتی شریف و مقاطع کارشناسی ارشد و دکتری به ترتیب در سال‌های ۱۳۶۶ و ۱۳۷۱ از دانشگاه سالفورد انگلستان در رشته کامپیوتر- نرم‌افزار دریافت نمود. دکتر پارسا از سال ۱۳۷۱ در دانشگاه علم و صنعت ایران در تهران مشغول به فعالیت گردید و هم‌اکنون دانشیار دانشکده مهندسی کامپیوتر و مدیر آزمایشگاه تحقیقاتی مهندسی معکوس این دانشگاه می‌باشد. زمینه‌های علمی مورد علاقه نام‌برده آزمون نرم‌افزار، مهندسی نرم‌افزار خودکار، کامپایلرها، مهندسی معکوس نرم‌افزار و موازی‌سازی خودکار برنامه‌های ترتیبی است. در حال حاضر پروژه‌های دانشجویان ارشد و دکتری ایشان روی مباحث تولید خودکار داده آزمون، مکان‌یابی آماری خطاهای پنهان، آزمون فازی و تعمیر خودکار برنامه‌ها است.