



دانشگاه صنعتی شریف  
دانشکده مهندسی کامپیوتر

گزارش پروژه ۸  
درس سیستم‌های عامل

## آزمون حافظه توسط برنامه EFI و بوت امن

نگارش

محمد داودآبادی

علیرضا صمیمی

پارسا علیزاده

امیرمحمد شاهرضایی

استاد راهنما

دکتر اسدی و دکتر جلیلی

شهریور ۱۴۰۳

## چکیده

در این پروژه بنا داریم تا در ابتدا با محیط برنامه نویسی EFI آشنا شویم به همین سبب کد Hello World با استفاده از EFI می‌نویسیم و گزارشی از آن ارائه می‌دهیم. در ادامه کمی در رابطه با آزمون حافظه، پیاده‌سازی آن و الگوریتم‌های آن تحقیق می‌کنیم. در آخر مقداری با Boot امن آشنا خواهیم شد و

...

کامل شه

# فهرست مطالب

۱	نوشتن برنامه World Hello با EFI	۱
۱	۱-۱ آماده کردن محیط برنامه نویسی EFI	۱
۱	۲-۱ تحلیل کد	۱
۱	۳-۱ اجرای کد	۱
۲	آزمون حافظه	۲
۳	۱-۲ آشنایی با آزمون حافظه	۳
۳	۲-۲ آزمون‌ها	۳
۳	۱-۲-۲ Walking Ones	۳
۳	۲-۲-۲ Identity	۳
۴	۳-۲-۲ Rowhammer	۴
۴	۴-۲-۲ DMA	۴
۴	۳-۲ اجرای آزمون حافظه	۴
۵	بوت امن	۵
۵	۱-۳ شبیه‌سازی	۵
۶	۲-۳ آماده‌سازی برنامه	۶
۶	۳-۳ آماده‌سازی بوت	۶
۶	۴-۳ بوت با اثرانگشت برنامه	۶

۷	۳-۵ بوت با امضای برنامه . . . . .
۷	۳-۶ منابع . . . . .
۱۱	۴ کارهای پیشین
۱۱	۴-۱ مسائل خوشه‌بندی . . . . .
۱۳	۴-۲ خوشه‌بندی $k$ -مرکز . . . . .
۱۵	۴-۳ مدل جویبار داده . . . . .
۱۶	۴-۴ تقریب‌پذیری . . . . .
۱۷	۵ نتایج جدید
۱۸	۶ نتیجه‌گیری
۱۹	مراجع
۲۱	واژه‌نامه
۲۳	آ مطالب تکمیلی

## فهرست جداول

۱-۴ نمونه‌هایی از کران پایین تقریب‌پذیری مسائل خوشه‌بندی ..... ۱۶

## فهرست تصاویر

۲	۱-۱
۴	۱-۲
۶	۱-۳ اضافه کردن sbat
۷	۲-۳ کپی کردن shim به ESP
۸	۳-۳ اضافه کردن boot option
۸	۴-۳ صفحه Mok Management
۹	۵-۳ وارد کردن امضای برنامه
۹	۶-۳ منوی انتخاب برنامه‌ها
۱۰	۷-۳ ساخت کلید
۱۰	۸-۳ امضای برنامه EFI
۱۳	۱-۴ نمونه‌ای از مسئله‌ی ۲- مرکز
۱۴	۲-۴ نمونه‌ای از مسئله‌ی ۲- مرکز با داده‌های پرت

# فصل ۱

## نوشتن برنامه World Hello با EFI

در این قسمت برای شروع کار یک برنامه‌ی World Hello می‌نویسیم.

### ۱-۱ آماده کردن محیط برنامه نویسی EFI

برای این قسمت ما از edk2 استفاده کرده‌ایم که می‌توانید آن را در این [لینک](#) مشاهده کنید. این ابزار یک سری کتابخانه برای نوشتن و ساختن برنامه‌ها و درایورهای EFI است.

### ۲-۱ تحلیل کد

کد این قسمت در HelloWorld.c نوشته شده است. تابع UefiMain تابعی است که در شروع برنامه اجرا می‌شود. در خط اول توسط تابع Print رشته مورد نظر را چاپ می‌کنیم و دقت می‌کنیم که در UEFI باید از رشته‌های عریض (کاراکترهای ۲ بایتی) استفاده شود.

### ۳-۱ اجرای کد

برای اجرای کد با استفاده از QEMU و OVMF یک محیط EFI بالا می‌آوریم تا برنامه را در آن اجرا کنیم. این کار با دادن OVMF به عنوان آپشن bios به QEMU به سادگی قابل انجام است. این اجرا در ۱-۱ قابل مشاهده است.

```
UEFI Interactive Shell v2.2
EDK II
UEFI v2.70 (EDK II, 0x00010000)
Mapping table
  FS0: Alias(s) :HD0a1::BLK1:
    PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)/HD(1,MBR,0xBE1AFDFA,0x3F,0xFBFC1)
  BLK0: Alias(s) :
    PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
  BLK2: Alias(s) :
    PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
Press ESC in 5 seconds to skip startup.nsh or any other key to continue.
Shell> fs:
'fs:' is not a valid mapping.
Shell> fs0:
FS0:\> hello.efi
Hello, world!
FS0:\> _
```

شکل ۱-۱



## فصل ۲

# آزمون حافظه

### ۱-۲ آشنایی با آزمون حافظه

آزمون حافظه به فرآیند تست کردن تایید کارکرد، درستی و کارایی حافظه سیستم گفته می‌شود. در این جا ما در فایل اصلی پروژه یعنی MemTest.C ۴ نوع تست آماده کرده‌ایم که هر کدام برای شرایطی خاص مناسب هستند.

### ۲-۲ آزمون‌ها

#### Walking Ones ۱-۲-۲

در این آزمون ما با یک الگو که شامل دقیقا یک بیت ۱ است شروع می‌کنیم و هر بار مقدار آن را یک شیفت دوری می‌دهیم. سپس در نهایت همه مقادیر را چک می‌کنیم. این آزمون کمک می‌کند مشکلات data bus پیدا شود. دلیل ۱ بودن دقیقا یک بیت این است که تفاوت یک بیت با دو بیت کناری شانس خطا را افزایش می‌دهد. این تست در تابع WalkingOnesTest پیاده‌سازی شده است.

#### Identity ۲-۲-۲

در این آزمون ما در هر خانه از حافظه آدرس خودش را می‌نویسیم. سپس در نهایت همه مقادیر را چک می‌کنیم. این آزمون بسیار ساده و کلی است و. در تابع IdentityTest پیاده‌سازی شده است.

```
UEFI Interactive Shell v2.2
EDK II
UEFI v2.70 (EDK II, 0x00010000)
Mapping table
FS0: Alias(s) :HD0a1:;BLK1:
    PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)/HD(1,MBR,0xBE1AFDFA,0x3F,0xFBFC1)
BLK0: Alias(s) :
    PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
BLK2: Alias(s) :
    PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
Press ESC in 4 seconds to skip startup.nsh or any other key to continue.
Shell> memtest
Total number of pages 512109
WalkingOnesTest passed
IdentityTest passed
RowhammerTest passed
DMATest passed
Shell> _
```

شکل ۱-۲

## Rowhammer ۳-۲-۲

در این آزمون ما حساس بودن حافظه به آسیب پذیری Rowhammer را می‌سنجیم. این آسیب پذیری سعی می‌کند با دسترسی با فرکانس بالا به چند سطر خاص در حافظه باعث شود مقادیر سطر دیگر عوض شوند. در این جا ما با استناد به این پیاده‌سازی از Rowhammer دو طرفه، تابع RowHammerTest را پیاده‌سازی کرده‌ایم.

## DMA ۴-۲-۲

در این آزمون ما درستی کارکرد زیرسامانه DMA برای انتقال داده بین دیسک و حافظه بدون دخالت پردازنده را می‌سنجیم. برای این کار در ابتدا یک فایل روی دیسک مورد نظر ساخته و با داده‌های خاص پر می‌شود. سپس این فایل در خانه‌هایی از حافظه خوانده شده و چک می‌شود که با چیزی که نوشته شده یکی باشد. این تست در تابع DMA پیاده‌سازی شده است.

## ۳-۲ اجرای آزمون حافظه

در ۱-۲ یک اجرا از تست‌های نوشته شده را نشان می‌دهیم.

## فصل ۳

### بوت امن

بوت امن (Secure Boot) یکی از ویژگی‌های امنیتی سیستم‌های UEFI است که هدف آن جلوگیری از اجرای کدهای غیرمجاز در فرآیند بوت سیستم می‌باشد. در این فرآیند، تمام نرم‌افزارهای بوت باید دارای امضای دیجیتال معتبر باشند. ابزار shim یک واسطه است که به کاربران امکان می‌دهد گواهی‌های سفارشی یا کلیدهای اضافی را برای بوت امن اضافه کنند، به خصوص در سیستم‌های لینوکسی که ممکن است امضاهای استاندارد میکروسافت را نداشته باشند. این روش باعث حفظ امنیت در عین انعطاف‌پذیری می‌شود.

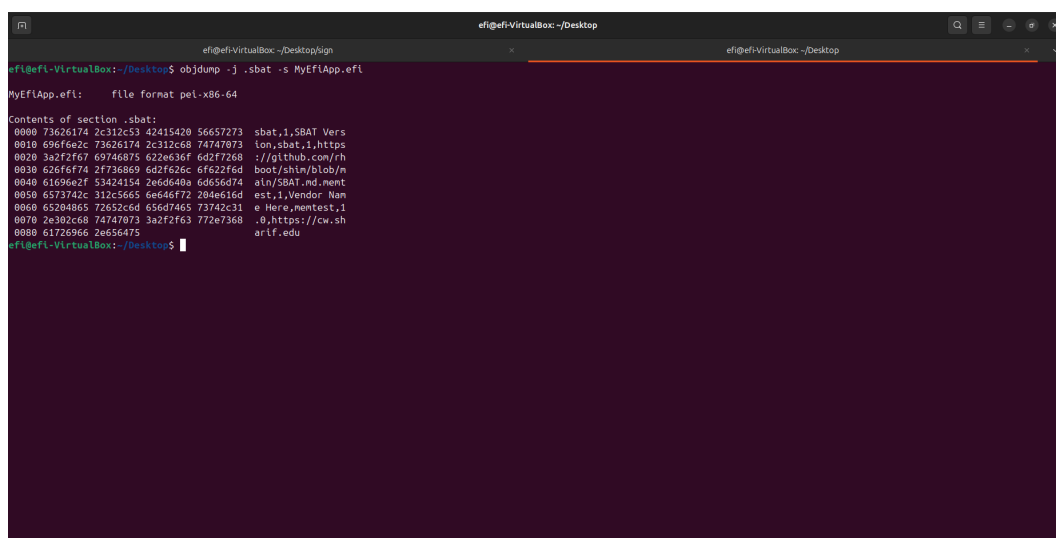
از آنجا که خروجی EFI ما امضای رسمی میکروسافت را ندارد، به shim برای اجرای آن در محیط بوت امن نیاز داریم.

### ۳-۱ شبیه‌سازی

برای شبیه‌سازی بوت امن از Virtualbox استفاده می‌کنیم. در این محیط می‌توانیم EFI و بوت امن را مشابه ماشین واقعی تست کنیم. یک سیستم عامل مجازی اوبونتو نصب می‌کنیم تا عملیات‌های مرتبط با راه‌اندازی برنامه را در آن انجام دهیم.

## ۲-۳ آماده‌سازی برنامه

باید به برنامه section به نام sbat اضافه کنیم. مهم است مقدار آن را shim بشناسد و به همین خاطر از بخش مشابهی در grub استفاده می‌کنیم.



```
efi@efi-VirtualBox: ~/Desktop
efi@efi-VirtualBox:~/Desktop$ objdump -j .sbat -s MyEfiApp.efi
MyEfiApp.efi:      file format pei-x86-64

Contents of section .sbat:
0000 73626174 2c312e53 42415420 56657273  sbat,1.SBAT Vers
0010 696f6e2c 73626174 2c312e68 74747073  lon,sbat,1.https
0020 3a2f2f67 69746875 622e636f 6d2f7268  ://github.com/rh
0030 626f6f74 2f736869 6d2f626c 6f622f6d  boot/shim/blob/n
0040 61696e2f 53424154 2e6d648a 6d656d74  aln/SBAT.md.ment
0050 6573742c 312c5665 6e646f72 204e616d  est,1.Vendor Nan
0060 65204865 72652c6d 656d7465 73742c31  e Here,mentest,1
0070 2e302c68 74747073 3a2f2f63 772e7368  .0,https://cw.sh
0080 61726966 2e656475  artf.edu
```

شکل ۳-۱: اضافه کردن sbat

## ۳-۳ آماده‌سازی بوت

برنامه shim به طور پیشفرض در اوبونتو نصب می‌شود. لازم است فایل‌های shim و برنامه EFI را به پارتیشن ESP منتقل می‌کنیم. اسم برنامه را به grubx64.efi تغییر می‌دهیم. به صورت پیشفرض shim برنامه‌ای با این اسم را لود می‌کند.

## ۴-۳ بوت با اثرانگشت برنامه

یک راه برای بوت امن این است که اجازه دهیم shim بوت شود و سپس اثرانگشت برنامه را به MokList اضافه کنیم. بعد از انجام این کار shim برنامه را اجرا می‌کند و خطای 0x1a Security Violation چاپ نمی‌شود. مراحل اضافه کردن اثرانگشت hash در ادامه نشان داده شده است.

```

efi@efi-VirtualBox: /boot/efi/EFI/test
efi@efi-VirtualBox: ~/Desktop/hign
efi@efi-VirtualBox: /boot/efi/EFI/test

0 upgraded, 2 newly installed, 0 to remove and 163 not upgraded.
Need to get 160 kB of archives.
After this operation, 623 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ir.archive.ubuntu.com/ubuntu noble/main amd64 liburing2 amd64 2.5-1build1 [21.1 kB]
Get:2 http://ir.archive.ubuntu.com/ubuntu noble/universe amd64 plocate amd64 1.1.19-2ubuntu2 [139 kB]
Fetched 160 kB in 1s (230 kB/s)
Selecting previously unselected package liburing2:amd64.
(Reading database ... 152705 files and directories currently installed.)
Preparing to unpack .../liburing2_2.5-1build1_amd64.deb ...
Unpacking liburing2:amd64 (2.5-1build1) ...
Selecting previously unselected package plocate.
Preparing to unpack .../plocate_1.1.19-2ubuntu2_amd64.deb ...
Unpacking plocate (1.1.19-2ubuntu2) ...
Setting up liburing2:amd64 (2.5-1build1) ...
Setting up plocate (1.1.19-2ubuntu2) ...
update-alternatives: using /usr/bin/plocate to provide /usr/bin/locate (locate) in auto mode
info: Selecting GID from range 100 to 999 ...
info: Adding group 'plocate' (GID 124) ...
Initializing plocate database; this may take some time... done
Created symlink /etc/systemd/system/timers.target.wants/plocate-updatedb.timer → /usr/lib/systemd/system/plocate-updatedb.timer.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu6.3) ...
efi@efi-VirtualBox: /boot/efi/EFI/test$ cp /usr/lib/shim/
BOOTX64.CSV
ls-not-revoked
mok/
shimx64.efi
shimx64.efi.dualsigned
shimx64.efi.signed.latest
shimx64.efi.signed.previous
shimx64.efi
mxm64.efi
shimx64.efi
cp: cannot create regular file './shimx64.efi': Permission denied
cp: cannot create regular file './mxm64.efi': Permission denied
efi@efi-VirtualBox: /boot/efi/EFI/test$ sudo cp /usr/lib/shim/(shimx64.efi,mxm64.efi) .
efi@efi-VirtualBox: /boot/efi/EFI/test$ ll
total 1600
drwxr-xr-x 2 root root 4096 Jan 29 14:11 ./
drwxr-xr-x 5 root root 4096 Jan 27 09:10 ../
-rwxr-xr-x 1 root root 13696 Jan 29 14:09 grubx64.efi*
-rwxr-xr-x 1 root root 856288 Jan 29 14:11 mxm64.efi*
-rwxr-xr-x 1 root root 857942 Jan 29 14:11 shimx64.efi*
efi@efi-VirtualBox: /boot/efi/EFI/test$

```

شکل ۳-۲: کپی کردن shim به ESP

## ۳-۵ بوت با امضای برنامه

راه دیگر برای بوت امن این است که یک کلید شخصی بسازیم و برنامه EFI را با آن امضا کنیم. بعد از امضا کردن و اجرای دوباره shim، می‌توانیم فایل DER امضا را به MokList اضافه کنیم. مزیت این روش این است که اگر برنامه بعداً تغییر کند فقط لازم است یک بار دیگر آن را امضا کنیم و دیگر طی کردن مراحل enroll key لازم نیست.

## ۳-۶ منابع

[https://wiki.archlinux.org/title/Unified\\_Extensible\\_Firmware\\_Interface/Secure\\_Boot](https://wiki.archlinux.org/title/Unified_Extensible_Firmware_Interface/Secure_Boot)

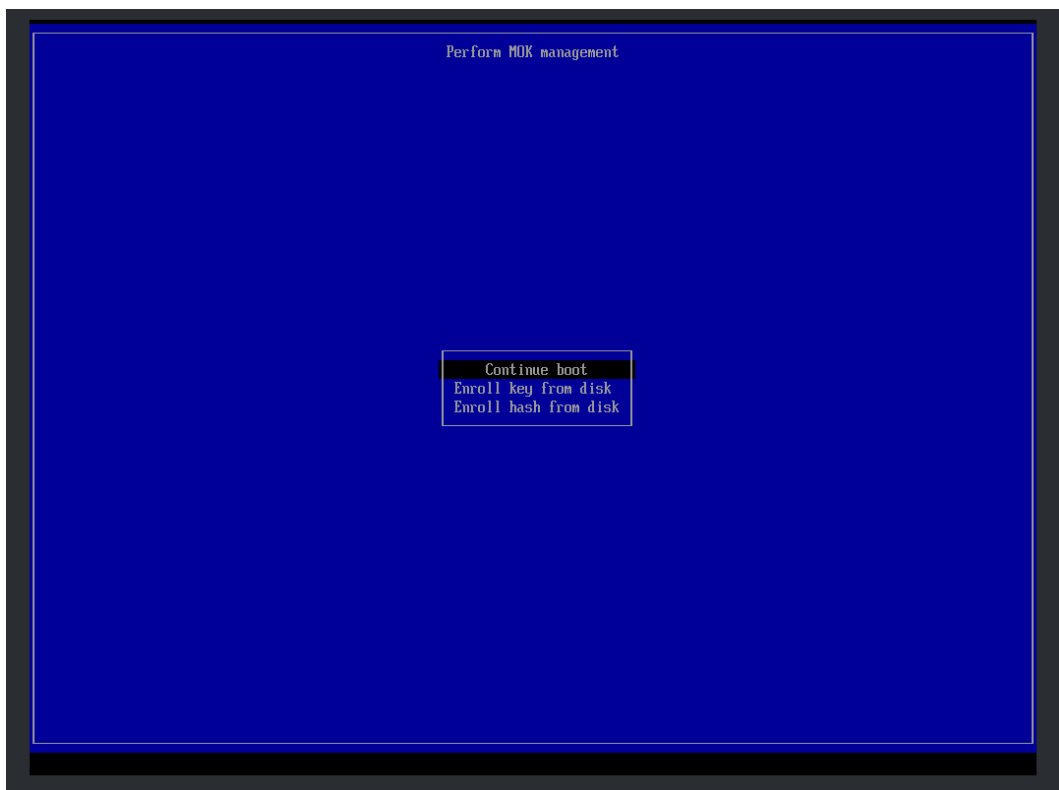
[https://www.funtoo.org/UEFI\\_Secure\\_Boot\\_and\\_SHIM](https://www.funtoo.org/UEFI_Secure_Boot_and_SHIM)

<https://github.com/rhboot/shim/blob/main/SBAT.md>

<https://github.com/rhboot/shim/issues/376>

```
efi@efi-VirtualBox:~$ efibootmgr --unicode --disk /dev/sdX --part Y --create --label "Shin" --loader ^C
efi@efi-VirtualBox:~$ efibootmgr --unicode --disk /dev/sda --part 1 --create --label "Mentest" --loader /EFI/test/shinx64.efi
Could not prepare Boot variable: Permission denied
efi@efi-VirtualBox:~$ sudo efibootmgr --unicode --disk /dev/sda --part 1 --create --label "Mentest" --loader /EFI/test/shinx64.efi
[sudo] password for efi:
BootCurrent: 0003
Timeout: 0 seconds
BootOrder: 0007,0003,0002,0001,0000,0004,0005
Boot0000* UEFI FvVol(7c8b9d9-8ab-4f34-aaea-3ee4f6516a1)/FvFile(462caa21-7614-4503-836e-8ab5f4662331)
Boot0001* UEFI VBOX CD-ROM VBOX2-01700376 PciRoot(0x0)/Pci(0x1,0x1)/Ata(1,0,0)
Boot0002* UEFI VBOX HARDISK VBOX2-01700376 PciRoot(0x0)/Pci(0xd,0x0)/Sata(0,65535,0)
Boot0003* Ubuntu HD(1,GPT,9146273e-8178-44d7-a556-4fc4a8be95c7,0x800,0x219800)/File(\EFI\ubuntu\shinx64.efi)
Boot0004* UEFI PXEvd (MAC:08002736d7ca) PciRoot(0x0)/Pci(0x3,0x0)/MAC(08002736d7ca,1)/IPv4(0.0.0.0,0.0.0.0)
Boot0005* UEFI PXEv6 (MAC:08002736d7ca) PciRoot(0x0)/Pci(0x3,0x0)/MAC(08002736d7ca,1)/IPv6([::]:[::]:[::]:[::]:[::]:[::]:[::]:[::])
Boot0006* MyEfiApp.efi PciRoot(0x0)/Pci(0xd,0x0)/Sata(0,65535,0)/HD(1,GPT,9146273e-8178-44d7-a556-4fc4a8be95c7,0x800,0x219800)/File(\EFI\test\MyEfiApp.efi)
Boot0007* Mentest HD(1,GPT,9146273e-8178-44d7-a556-4fc4a8be95c7,0x800,0x219800)/File(\EFI\test\shinx64.efi)
efi@efi-VirtualBox:~$
```

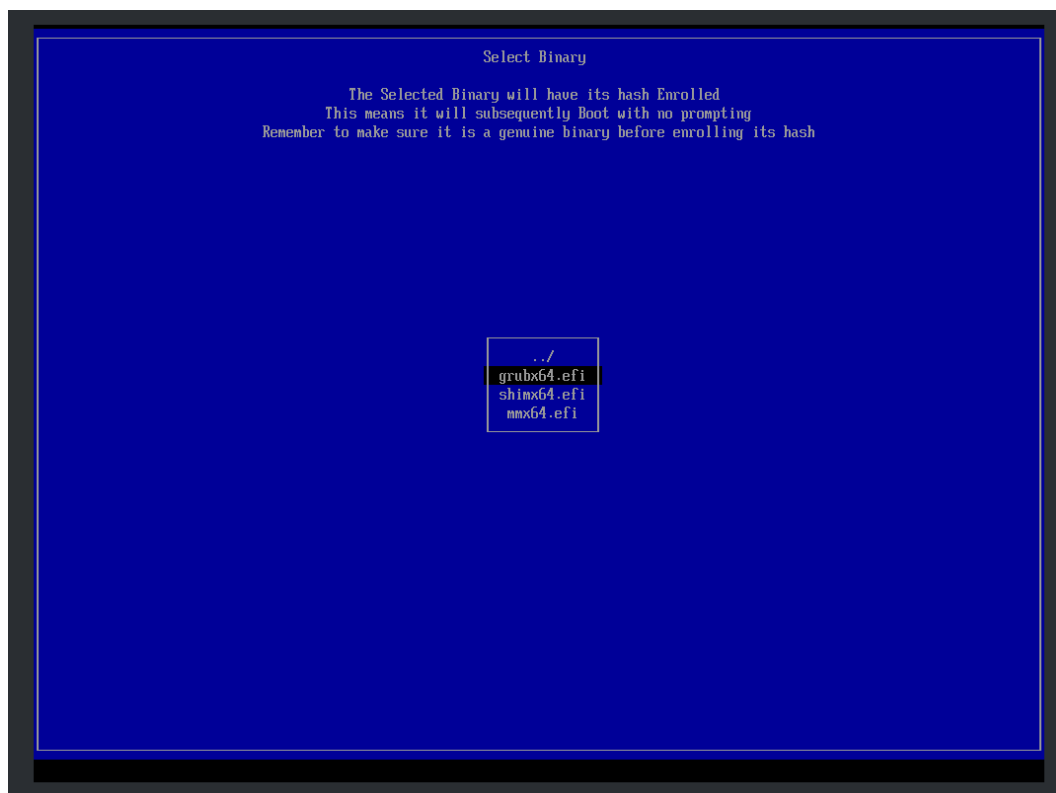
شکل ۳-۳: اضافه کردن boot option



شکل ۳-۴: صفحه Mok Management



شکل ۳-۵: وارد کردن امضای برنامه



شکل ۳-۶: منوی انتخاب برنامه‌ها





## فصل ۴

# کارهای پیشین

در فصل سوم پایان نامه، کارهای پیشین انجام شده روی مسئله به تفصیل توضیح داده می شود. نمونه ای از فصل کارهای پیشین در زیر آمده است.<sup>۱</sup>

### ۴-۱ مسائل خوشه بندی

مسئله ی خوشه بندی<sup>۲</sup> یکی از مهم ترین مسائل در زمینه ی داده کاوی به حساب می آید. در این مسئله، هدف دسته بندی تعدادی شیء به گونه ای است که اشیاء درون یک دسته (خوشه)، نسبت به یکدیگر در برابر دسته های دیگر شبیه تر باشند (معیارهای متفاوتی برای تشابه تعریف می گردد). این مسئله در حوزه های مختلفی از علوم کامپیوتر از جمله داده کاوی، جست و جوی الگو<sup>۳</sup>، پردازش تصویر<sup>۴</sup>، بازیابی اطلاعات<sup>۵</sup> و رایانش زیستی<sup>۶</sup> مورد استفاده قرار می گیرد [۱].

تا کنون راه حل های زیادی برای این مسئله ارائه شده است که از لحاظ معیار تشخیص خوشه ها و نحوه ی انتخاب یک خوشه، با یکدیگر تفاوت بسیاری دارند. به همین خاطر مسئله ی خوشه بندی یک مسئله ی بهینه سازی چند هدفه<sup>۷</sup> محسوب می شود.

همان طور که در مرجع [۲] ذکر شده است، خوشه در خوشه بندی تعریف واحدی ندارد و یکی از

---

<sup>۱</sup> مطالب این فصل نمونه از پایان نامه ی آقای بهنام حاتمی گرفته شده است.

<sup>۲</sup> Clustering

<sup>۳</sup> Pattern recognition

<sup>۴</sup> Image analysis

<sup>۵</sup> Information retrieval

<sup>۶</sup> Bioinformatics

<sup>۷</sup> Multi-objective

دلایل وجود الگوریتم‌های متفاوت، همین تفاوت تعریف‌ها از خوشه است. بنابراین با توجه به مدلی که برای خوشه‌ها ارائه می‌شود، الگوریتم متفاوتی نیز ارائه می‌گردد. در ادامه به بررسی تعدادی از معروف‌ترین مدل‌های مطرح می‌پردازیم:

- **مدل‌های مرکزگرا:** در این مدل‌ها، هر دسته با یک مرکز نشان داده می‌شود. از جمله معروف‌ترین روش‌های خوشه‌بندی بر اساس این مدل، خوشه‌بندی  $k$ -مرکز، خوشه‌بندی  $k$ -میانگین<sup>۸</sup> و خوشه‌بندی  $k$ -میان<sup>۹</sup> است.

- **مدل‌های مبتنی بر توزیع نقاط:** در این مدل، دسته‌ها با فرض پیروی از یک توزیع احتمالی مشخص می‌شوند. از جمله الگوریتم‌های معروف ارائه شده در این مدل، الگوریتم بیشینه‌سازی امید ریاضی<sup>۱۰</sup> است.

- **مدل‌های مبتنی بر تراکم نقاط:** در این مدل، خوشه‌ها متناسب با ناحیه‌های متراکم نقاط در مجموعه داده مورد استفاده قرار می‌گیرد.

- **مدل‌های مبتنی بر گراف:** در این مدل، هر خوشه به مجموعه از رئوس گفته می‌شود که تمام رئوس آن با یک‌دیگر همسایه باشند. از جمله الگوریتم‌های معروف این مدل، الگوریتم خوشه‌بندی HCS<sup>۱۱</sup> است.

الگوریتم‌های ارائه شده تنها از نظر نوع مدل با یک‌دیگر متفاوت نیستند. بلکه، می‌توان آن‌ها را از لحاظ نحوه‌ی تخصیص نقاط بین خوشه‌ها نیز تقسیم‌بندی کرد:

- **تخصیص قطعی داده‌ها:** در این نوع خوشه‌بندی هر داده دقیقاً به یک خوشه اختصاص داده می‌شود.

- **تخصیص قطعی داده‌ها با داده‌ی پرت:** در این نوع خوشه‌بندی ممکن است بعضی از داده‌ها به هیچ خوشه‌ای اختصاص نیابد، اما بقیه داده‌ها هر کدام دقیقاً به یک خوشه اختصاص می‌یابد.

- **تخصیص قطعی داده:** در این نوع خوشه‌بندی هر داده دقیقاً به یک خوشه اختصاص داده می‌شود.

- **خوشه‌بندی هم‌پوشان:** در این نوع خوشه‌بندی هر داده می‌تواند به چند خوشه اختصاص داده شود. در گونه‌ای از این مدل، می‌توان هر نقطه را با احتمالی به هر خوشه اختصاص می‌یابد. به این گونه از خوشه‌بندی، خوشه‌بندی نرم<sup>۱۲</sup> گفته می‌شود.

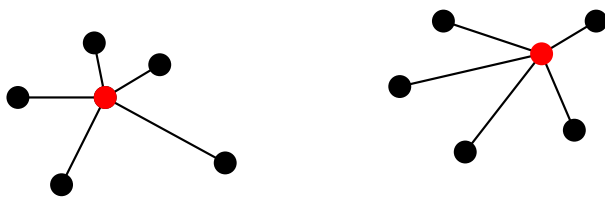
<sup>۸</sup>  $k$ -Means

<sup>۹</sup>  $k$ -Median

<sup>۱۰</sup> Expectation-maximization

<sup>۱۱</sup> Highly Connected Subgraphs

<sup>۱۲</sup> Soft clustering



شکل ۴-۱: نمونه‌ای از مسئله‌ی ۲-مرکز

• خوشه‌بندی سلسه‌مراتبی: در این نوع خوشه‌ها، داده‌ها به گونه‌ای به خوشه‌ها تخصیص داده می‌شود که دو خوشه یا اشتراک ندارند یا یکی به طور کامل دیگری را می‌پوشاند. در واقع در بین خوشه‌ها، رابطه‌ی پدر فرزندی برقرار است.

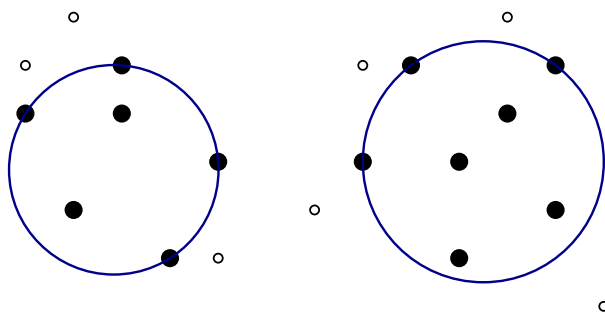
در بین دسته‌بندی‌های ذکر شده، تمرکز اصلی این پایان‌نامه بر روی مدل مرکزگرا و خوشه‌بندی قطعی با داده‌های پرت با مدل  $k$ -مرکز است. همان‌طور که ذکر شد علاوه بر مسئله‌ی  $k$ -مرکز که به تفصیل مورد بررسی قرار می‌گیرد،  $k$ -میانه و  $k$ -میانگین از جمله معروف‌ترین خوشه‌بندی‌های مدل مرکزگرا هستند. در خوشه‌بندی  $k$ -میانه، هدف افراز نقاط به  $k$  خوشه است به گونه‌ای که مجموع مربع فاصله‌ی هر نقطه از میانه‌ی نقاط آن خوشه، کمینه گردد. در خوشه‌بندی  $k$ -میانگین، هدف افراز نقاط به  $k$  خوشه است به گونه‌ای که مجموع فاصله‌ی هر نقطه از میانگین نقاط داخل خوشه (یا مرکز آن خوشه) کمینه گردد.

## ۴-۲ خوشه‌بندی $k$ -مرکز

یکی از رویکردهای شناخته‌شده برای مسئله‌ی خوشه‌بندی، مسئله‌ی  $k$ -مرکز است. در این مسئله هدف، پیدا کردن  $k$  نقطه به عنوان مرکز دسته‌ها است به‌طوری‌که شعاع دسته‌ها تا حد ممکن کمینه شود. مثالی از مسئله‌ی ۲-مرکز در شکل ۴-۱ نشان داده شده است. در این پژوهش، مسئله‌ی  $k$ -مرکز با متریک‌های خاص و برای  $k$ های کوچک مورد بررسی قرار گرفته است و هر کدام از تعریف رسمی مسئله‌ی  $k$ -مرکز در زیر آمده است:

**مسئله‌ی ۴-۱ ( $k$ -مرکز)** گراف کامل بدون جهت  $G = (V, E)$  با تابع فاصله‌ی  $d$ ، که از نامساوی مثلثی پیروی می‌کند داده شده است. زیرمجموعه‌ی  $S \subseteq V$  با اندازه‌ی  $k$  را به گونه‌ای انتخاب کنید که عبارت زیر را کمینه کند:

$$\max_{v \in V} \{ \min_{s \in S} d(v, s) \} \quad (۴-۱)$$



شکل ۲-۴: نمونه‌ای از مسئله‌ی ۲-مرکز با داده‌های پرت

گونه‌های مختلفی از مسئله‌ی  $k$ -مرکز با محدودیت‌های متفاوت توسط پژوهشگران مورد مطالعه قرار گرفته است. از جمله‌ی این گونه‌ها، می‌توان به حالتی که در بین داده‌های ورودی، داده‌های پرت وجود دارد، اشاره کرد. در واقع در این مسئله، قبل از خوشه‌بندی می‌توانیم تعدادی از نقاط ورودی را حذف نموده و سپس به خوشه‌بندی نقاط پردازیم. سختی این مسئله از آنجاست که نه تنها باید مسئله‌ی خوشه‌بندی را حل نمود، بلکه در ابتدا باید تصمیم گرفت که کدام یک از داده‌ها را به عنوان داده‌ی پرت در نظر گرفت که بهترین جواب در زمان خوشه‌بندی به دست آید. در واقع اگر تعداد نقاط پرتی که مجاز به حذف است، برابر صفر باشد، مسئله به مسئله‌ی  $k$ -مرکز تبدیل می‌شود. نمونه‌ای از مسئله‌ی ۲-مرکز با ۷ داده‌ی پرت را در شکل ۲-۴ می‌توانید ببینید. تعریف دقیق‌تر این مسئله در زیر آمده است:

**مسئله‌ی ۲-۴ ( $k$ -مرکز با داده‌های پرت)** یک گراف کامل بدون جهت  $G = (V, E)$  با تابع فاصله‌ی  $d$ ، که از نامساوی مثلثی پیروی می‌کند داده شده است. زیرمجموعه‌ی  $Z \subseteq V$  با اندازه‌ی  $z$  و مجموعه‌ی  $S \subseteq V - Z$  با اندازه‌ی  $k$  را انتخاب کنید به طوری که عبارت زیر را کمینه کند:

$$\max_{v \in V-Z} \{ \min_{s \in S} d(v, s) \} \quad (2-4)$$

گونه‌ی دیگری از مسئله‌ی  $k$ -مرکز که در سال‌های اخیر مورد توجه قرار گرفته است، حالت جویبار داده‌ی آن است. در این گونه از مسئله‌ی  $k$ -مرکز، در ابتدا تمام نقاط در دسترس نیستند، بلکه به مرور زمان نقاط در دسترس قرار می‌گیرند. محدودیت دومی که وجود دارد، محدودیت حافظه است، به طوری که نمی‌توان تمام نقاط را در حافظه نگه داشت و بعضاً حتی امکان نگه‌داری در حافظه‌ی جانبی نیز وجود ندارد و به طور معمول باید مرتبه‌ی حافظه‌ای کم‌تر از مرتبه حافظه‌ی خطی<sup>۱۳</sup> متناسب با تعداد نقاط استفاده نمود. از این به بعد به چنین مرتبه‌ای، مرتبه‌ی زیرخطی<sup>۱۴</sup> می‌گوییم. مدلی که ما در این پژوهش بر روی آن تمرکز داریم مدل جویبار داده تک‌گذره<sup>۱۵</sup> [۳] است. یعنی تنها یک بار می‌توان از ابتدا تا انتهای داده‌ها را بررسی کرد و پس

<sup>۱۳</sup> Linear  
<sup>۱۴</sup> sublinear  
<sup>۱۵</sup> Single pass

از عبور از یک داده، اگر آن داده در حافظه ذخیره نشده باشد، دیگر به آن دسترسی وجود ندارد. علاوه بر این، در هر لحظه باید بتوان به پرسمان (برای تمام نقاطی از جویبار داده که تاکنون به آن دسترسی داشته‌ایم) پاسخ داد.

**مسئله ۳-۴** ( $k$ -مرکز در حالت جویبار داده) مجموعه‌ای از نقاط در فضای  $d$ -بعدی به مرور زمان داده می‌شود. در هر لحظه از زمان، به ازای مجموعه‌ی  $U$  از نقاطی که تا کنون وارد شده‌اند، زیرمجموعه‌ی  $S \subseteq U$  با اندازه‌ی  $k$  را انتخاب کنید به طوری که عبارت زیر کمینه شود:

$$\max_{u \in U} \{ \min_{s \in S} d(u, s) \} \quad (3-4)$$

از آنجایی که گونه‌ی جویبار داده و داده پرت مسئله‌ی  $k$ -مرکز به علت به‌روز بودن مبحث داده‌های حجیم<sup>۱۶</sup>، به تازگی مورد توجه قرار گرفته است. در این تحقیق سعی شده است که تمرکز بر روی این گونه‌ی خاص از مسئله باشد. همچنین در این پژوهش سعی می‌شود گونه‌های مسئله را برای انواع متریک‌ها و برای  $k$ های کوچک نیز مورد بررسی قرار داد.

## ۳-۴ مدل جویبار داده

همان‌طور که ذکر شد مسئله‌ی  $k$ -مرکز در حالت داده‌های پرت و جویبار داده، گونه‌های تعمیم‌یافته از مسئله‌ی  $k$ -مرکز هستند و در حالت‌های خاص به مسئله‌ی  $k$ -مرکز کاهش پیدا می‌کنند. مسئله‌ی  $k$ -مرکز در حوزه‌ی مسائل ان‌پی-سخت<sup>۱۷</sup> قرار می‌گیرد و با فرض  $P \neq NP$  الگوریتم دقیق با زمان چندجمله‌ای برای آن وجود ندارد [۴]. بنابراین برای حل کارای<sup>۱۸</sup> این مسائل از الگوریتم‌های تقریبی<sup>۱۹</sup> استفاده می‌شود.

برای مسئله‌ی  $k$ -مرکز، دو الگوریتم تقریبی معروف وجود دارد. در الگوریتم اول، که به روش حریصانه<sup>۲۰</sup> عمل می‌کند، در هر مرحله بهترین مرکز ممکن را انتخاب می‌کند به طوری تا حد ممکن از مراکز قبلی دور باشد [۵]. این الگوریتم، الگوریتم تقریبی با ضریب تقریب ۲ ارائه می‌دهد. در الگوریتم دوم، با استفاده از مسئله‌ی مجموعه‌ی غالب کمینه<sup>۲۱</sup>، الگوریتمی با ضریب تقریب ۲ ارائه می‌گردد [۶]. همچنین ثابت شده است، که بهتر از این ضریب تقریب، الگوریتمی نمی‌توان ارائه داد مگر آن‌که  $P = NP$  باشد.

<sup>۱۶</sup> Big data

<sup>۱۷</sup> NP-hard

<sup>۱۸</sup> Efficient

<sup>۱۹</sup> Approximation algorithm

<sup>۲۰</sup> Greedy

<sup>۲۱</sup> Dominating set

جدول ۴-۱: نمونه‌هایی از کران پایین تقریب‌پذیری مسائل خوشه‌بندی

مسئله	کران پایین تقریب‌پذیری
$k$ - مرکز	۲ [۶]
$k$ - مرکز در فضای اقلیدسی	۱/۸۲۲ [۱۵]
۱ - مرکز در حالت جویبار داده	$\frac{1+\sqrt{2}}{4}$ [۱۱]
$k$ - مرکز با نقاط پرت و نقاط اجباری	۳ [۱۰]

برای مسئله‌ی  $k$  - مرکز در حالت جویبار داده برای ابعاد بالا، بهترین الگوریتم موجود ضریب تقریب  $2 + \varepsilon$  دارد [۷، ۸، ۹] و ثابت می‌شود الگوریتمی با ضریب تقریب بهتر از ۲ نمی‌توان ارائه داد. برای مسئله‌ی  $k$  - مرکز با داده‌ی پرت در حالت جویبار داده نیز، بهترین الگوریتم ارائه شده، الگوریتمی با ضریب تقریب  $4 + \varepsilon$  است که با کران پایین ۳ هنوز اختلاف قابل توجهی دارد [۱۰].

برای  $k$  های کوچک به خصوص،  $k = 1, 2$ ، الگوریتم‌های بهتری ارائه شده است. بهترین الگوریتم ارائه شده برای مسئله‌ی ۱ - مرکز در حالت جویبار داده برای ابعاد بالا، دارای ضریب تقریب  $1/22$  است و کران پایین  $\frac{1+\sqrt{2}}{4}$  نیز برای این مسئله اثبات شده است [۱۱، ۱۲]. برای مسئله ۲ - مرکز در حالت جویبار داده برای ابعاد بالا، اخیراً راه‌حلی با ضریب تقریب  $1/8 + \varepsilon$  ارائه شده است [۱۳]. برای مسئله‌ی ۱ - مرکز با داده‌ی پرت، تنها الگوریتم موجود، الگوریتمی با ضریب تقریب  $1/73$  است [۱۴].

## ۴-۴ تقریب‌پذیری

یکی از راه‌کارهایی که برای کارآمد کردن راه‌حل ارائه شده برای یک مسئله وجود دارد، استفاده از الگوریتم‌های تقریبی برای حل آن مسئله است. یکی از عمده‌ترین دغدغه‌های مطرح در الگوریتم‌های تقریبی کاهش ضریب تقریب است. در بعضی از موارد حتی امکان ارائه‌ی الگوریتم تقریبی با ضریبی ثابت نیز وجود ندارد. به طور مثال، الگوریتم تقریبی با ضریب تقریب کم‌تر از ۲، برای مسئله‌ی  $k$  - مرکز وجود ندارد مگر این‌که  $P = NP$  باشد. برای مسائل مختلف، معمولاً می‌توان کران پایینی برای میزان تقریب‌پذیری آن‌ها ارائه داد. در واقع برای برخی مسائل ان‌پی-سخت، علاوه بر این که الگوریتم کارآمدی وجود ندارد، بعضاً الگوریتم تقریبی با ضریبی تقریب کم و نزدیک به یک نیز وجود ندارد. در جدول ۴-۱ میزان تقریب‌پذیری مسائل مختلفی که در این پایان‌نامه مورد استفاده قرار می‌گیرد را می‌بینید.

## فصل ۵

### نتایج جدید

در این فصل نتایج جدید به دست آمده در پایان نامه توضیح داده می شود. در صورت نیاز می توان نتایج جدید را در قالب چند فصل ارائه نمود. همچنین در صورت وجود پیاده سازی، بهتر است نتایج پیاده سازی را در فصل مستقلی پس از این فصل قرار داد.

## فصل ۶

### نتیجه‌گیری

در این فصل، ضمن جمع‌بندی نتایج جدید ارائه‌شده در پایان‌نامه یا رساله، مسائل باز باقی‌مانده و همچنین پیشنهادهایی برای ادامه‌ی کار ارائه می‌شوند.



# Bibliography

- [1] J. Han and M. Kamber. *Data Mining, Southeast Asia Edition: Concepts and Techniques*. Morgan kaufmann, 2006.
- [2] V. Estivill-Castro. Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, 4(1):65–75, 2002.
- [3] C. C. Aggarwal. *Data streams: models and algorithms*. Springer Science & Business Media, 2007.
- [4] M. R. Garey and D. S. Johnson. Computers and intractability: a guide to the theory of NP-completeness. *Freeman & Co.*, 1979.
- [5] N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13(1):182–196, 1984.
- [6] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag New York, Inc., 2001.
- [7] R. M. McCutchen and S. Khuller. Streaming algorithms for k-center clustering with outliers and with anonymity. In *Proceedings of the 11th International Workshop on Approximation Algorithms*, pages 165–178, 2008.
- [8] S. Guha. Tight results for clustering and summarizing data streams. In *Proceedings of the 12th International Conference on Database Theory*, pages 268–275, 2009.
- [9] H.-K. Ahn, H.-S. Kim, S.-S. Kim, and W. Son. Computing k centers over streaming data for small k. *International Journal of Computational Geometry and Applications*, 24(02):107–123, 2014.
- [10] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan. Algorithms for facility location problems with outliers. In *Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms*, pages 642–651, 2001.

- [11] P. K. Agarwal and R. Sharathkumar. Streaming algorithms for extent problems in high dimensions. In *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms*, pages 1481–1489, 2010.
- [12] T. M. Chan and V. Pathak. Streaming and dynamic algorithms for minimum enclosing balls in high dimensions. *Computational Geometry: Theory and Applications*, 47(2):240–247, 2014.
- [13] S.-S. Kim and H.-K. Ahn. An improved data stream algorithm for clustering. In *Proceedings of the 11th Latin American Symposium on Theoretical Informatics*, pages 273–284, 2014.
- [14] H. Zarrabi-Zadeh and A. Mukhopadhyay. Streaming 1-center with outliers in high dimensions. In *Proceedings of the 21st Canadian Conference on Computational Geometry*, pages 83–86, 2009.
- [15] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, pages 296–345. PWS Publishing Co., 1997.

# واژه‌نامه

## الف

experimental . . . . . تجربی	heuristic . . . . . ابتکاری
density . . . . . تراکم	high dimensions . . . . . ابعاد بالا
approximation . . . . . تقریب	bias . . . . . اریب
partition . . . . . تقسیم‌بندی	threshold . . . . . آستانه
mesh . . . . . توری	pigeonhole principle . . . . . اصل لانه‌ی کبوتری
distributed . . . . . توزیع‌شده	NP-Hard . . . . . ان‌پی-سخت
	transition . . . . . انتقال

## ت

## ج

separable . . . . . جداپذیر
black box . . . . . جعبه سیاه
data stream . . . . . جویبار داده

## ح

extreme . . . . . حدی
greedy . . . . . حریصانه

## خ

cluster . . . . . خوشه
linear . . . . . خطی

## ب

online . . . . . برخط
linear programming . . . . . برنامه‌ریزی خطی
optimum . . . . . بهینه
maximum . . . . . بیشینه

## پ

outlier . . . . . پرت
query . . . . . پرسمان
cover . . . . . پوشش
complexity . . . . . پیچیدگی

## د

data ..... داده  
 data mining ..... داده کاوی  
 outlier data ..... داده‌ی پرت  
 doubling ..... دوبرابر سازی  
 binary ..... دودویی

## ف

distance ..... فاصله  
 space ..... فضا

## ق

deterministic ..... قطعی

## ر

vertex ..... رأس  
 formal ..... رسمی

## ک

efficient ..... کارا  
 candidate ..... کاندیدا  
 minimum ..... کمینه

## ز

sublinear ..... زیرخطی

## م

set ..... مجموعه  
 coreset ..... مجموعه هسته  
 planar ..... سطح  
 parallelization ..... موازی سازی  
 buffer ..... میان گیر

## س

amortized ..... سرشکن  
 hierarchichal ..... سلسه مراتبی

## ش

pseudocode ..... شبه کد  
 object ..... شیء

## ن

inversion ..... نابه جایی  
 invariant ..... ناورد  
 center point ..... نقطه‌ی مرکزی  
 half space ..... نیم فضا

## ص

satisfiability ..... صدق پذیری

## ه

price of anarchy (POA) ..... هزینه‌ی آشوب

## غ

dominate ..... غلبه

## ی

edge ..... یال

پیوست آ

مطالب تکمیلی

پیوست‌های خود را در صورت وجود می‌توانید در این قسمت قرار دهید.

## **Abstract**

We present a standard template for typesetting theses in Persian. The template is based on the `XYLATEX Persian` package for the `LATEX` typesetting system. This write-up shows a sample usage of this template.

**Keywords:** Thesis, Typesetting, Template, `XYLATEX Persian`



Sharif University of Technology  
Department of Computer Engineering

B.Sc. Thesis

# **A Standard Template for Typesetting Theses in Persian**

By:

**The Author**

Supervisor:

**Dr. Supervisor**

September 2024