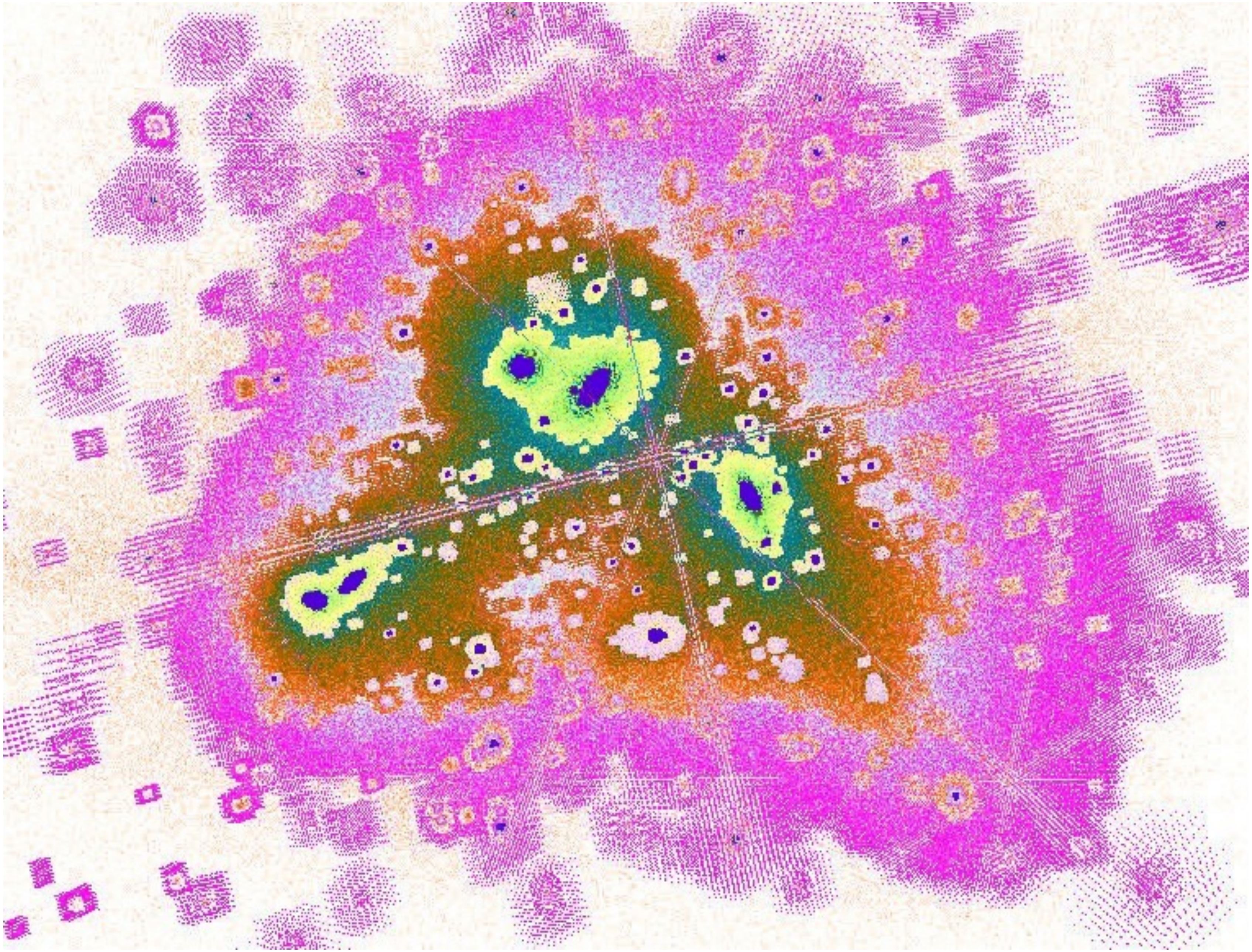
User's Guide

version 1.0 (build 098)



AHF - Amiga's Halo Finder

Alexander Knebe, Feb 2014

DISCLAIMER

All Software described in this User's Guide and provided for download comes with no warranty or guarantee to function!

Further, this Guide does not claim to be complete; it has been compiled to the best knowledge and primarily lists those options and features that are considered "useful" for the general black-box user...

The proper references for all things ***AHF*** are the code papers

Gill S.P.D., Knebe A., Gibson B.K., 2004, MNRAS, 351, 399
Knollmann S.R., Knebe A., 2009, ApJS, 182,608

Please refer to these publications for more information and the relevant tests and please
cite them both when publishing results based upon ***AHF***.

Some additional nice articles to consider reading are:

Knebe et al., 2011, MNRAS, 415, 2293 (Haloes going MAD paper) ,
Onions et al., 2012, MNRAS, in press (Subhaloes going Notts paper)

Introduction

- **MLAPM** (**M**ulti-**L**evel-**A**daptive-**P**article-**M**esh)

when	what	who
1997	grid structure	Andrew Green
2000	complete revision	Alexander Knebe
2001	public release	Knebe, Green & Binney (2001)
2002	software package for lightcones	Enn Saar
2004	**MHF**: on-the-fly halo identification	Stuart Gill (Gill, Knebe & Gibson 2004)
2005	name change **MLAPM** –> ***AMIGA***	Alexander Knebe

- **MLAPM** (**M**ulti-**L**evel-**A**daptive-**P**article-**M**esh)

when	**what**	**who**
1997	grid structure	Andrew Green
2000	complete revision	Alexander Knebe
2001	public release	Knebe, Green & Binney (2001)
2002	software package for lightcones	Enn Saar
2004	**MHF**: on-the-fly halo identification	Stuart Gill (Gill, Knebe & Gibson 2004)
2005	name change **MLAPM** –> ***AMIGA***	Alexander Knebe

MLAPM's users-guide.pdf **already contains information about MHF!**

- ***AMIGA*** ***(Adaptive Mesh Investigations of Galaxy Assembly)***

when	**what**	**who**
2005	name change **MLAPM** –> ***AMIGA***	Alexander Knebe
	name change **MHF** –> ***AHF***	Alexander Knebe
2007	- release of MPI enabled ***AHF***	Steffen Knollmann
	- revisions over revisions...	Alexander Knebe, Steffen Knollmann, Kristin Warnick, Claudio Llinares, ...

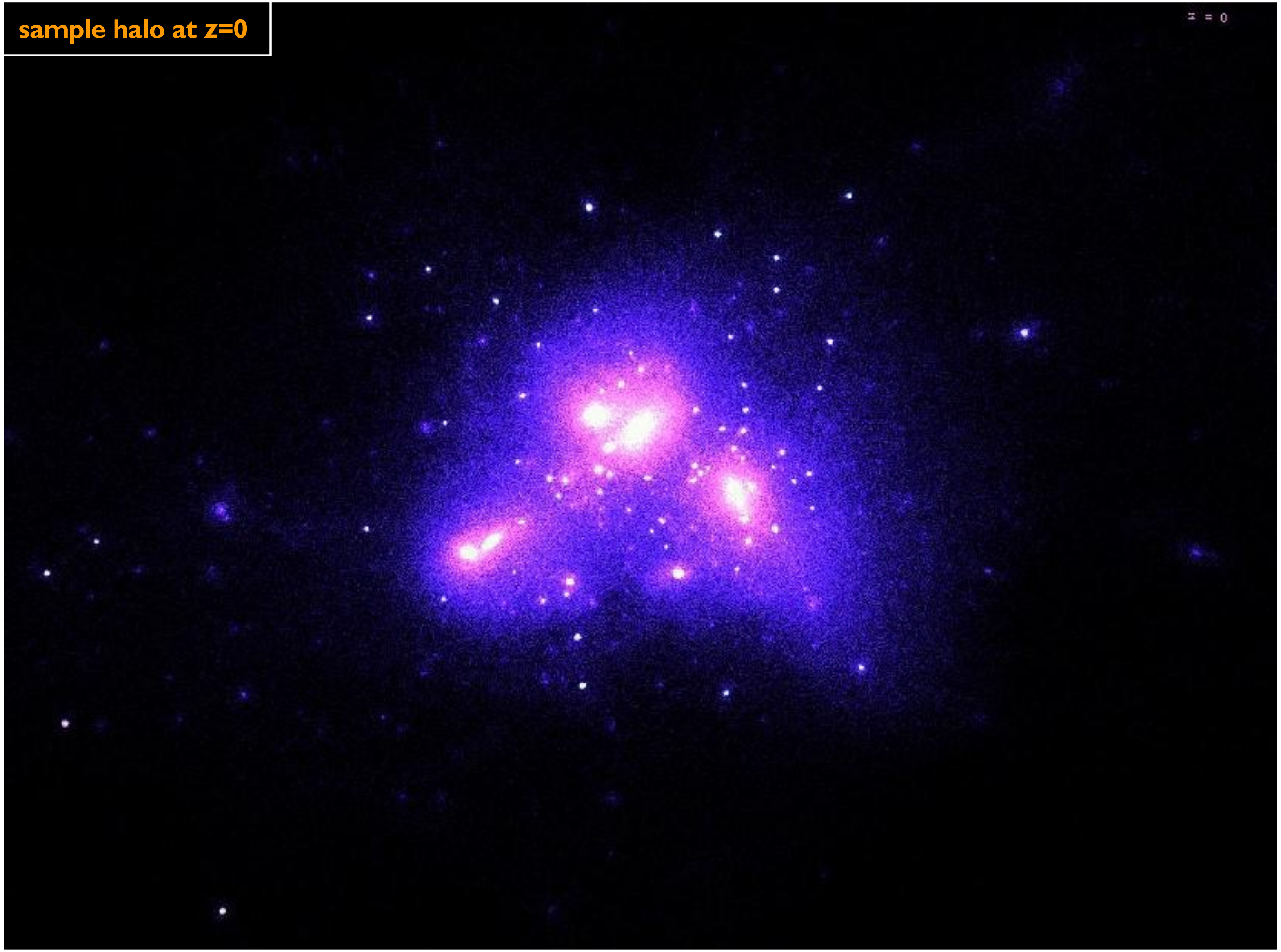
- ***AHF (still Adaptive Mesh Investigations of Galaxy Assembly)***

when	what	who
2012	- separation of simulation code ***AMIGA*** and halo finder code ***AHF***	Alexander Knebe
	- new user interface	Steffen Knollmann

Concept

Concept

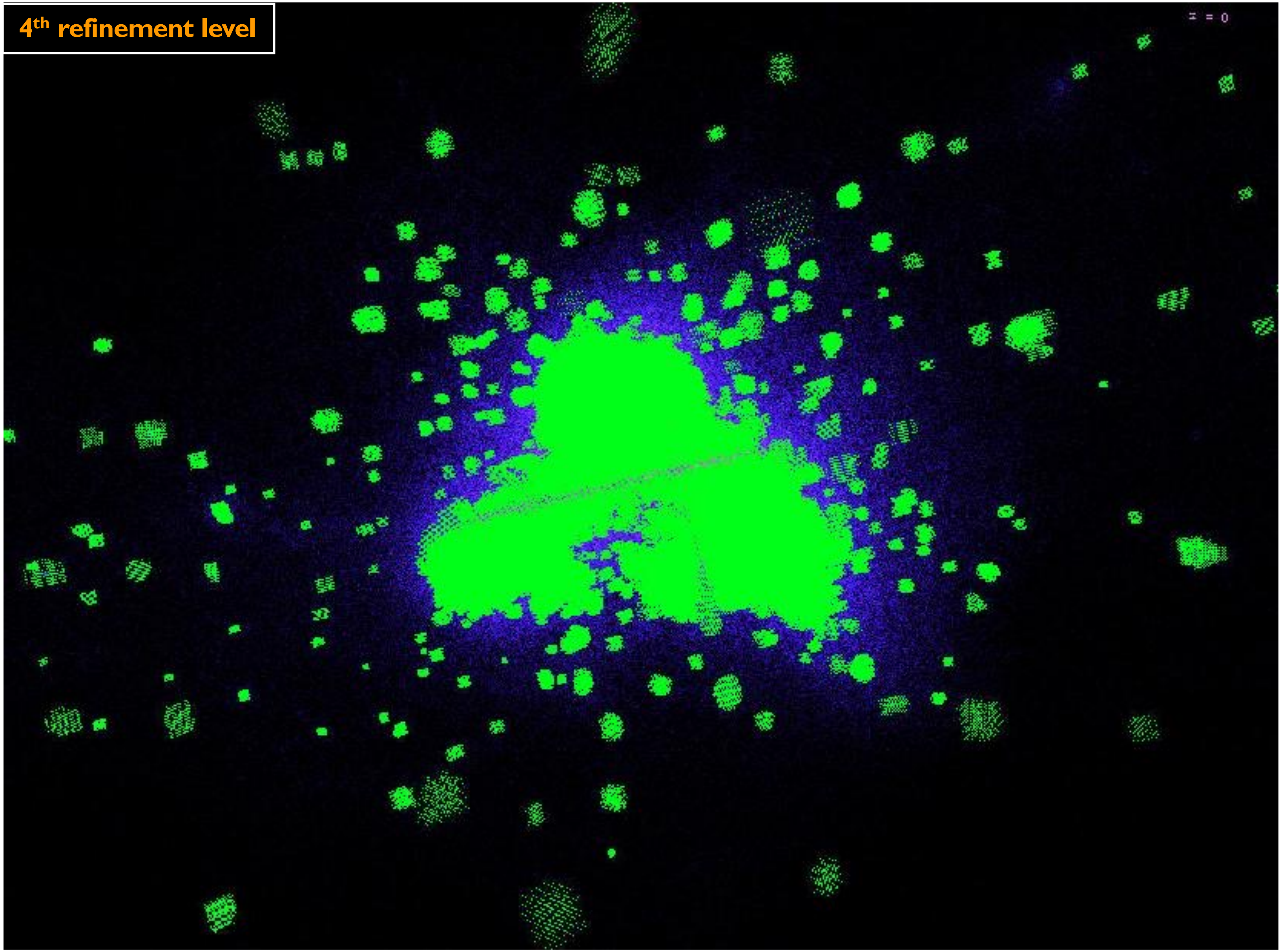
- finding prospective halo centres
- collecting particles possibly bound to centre
- removing unbound particles
- calculating halo properties


sample halo at z=0
z = 0

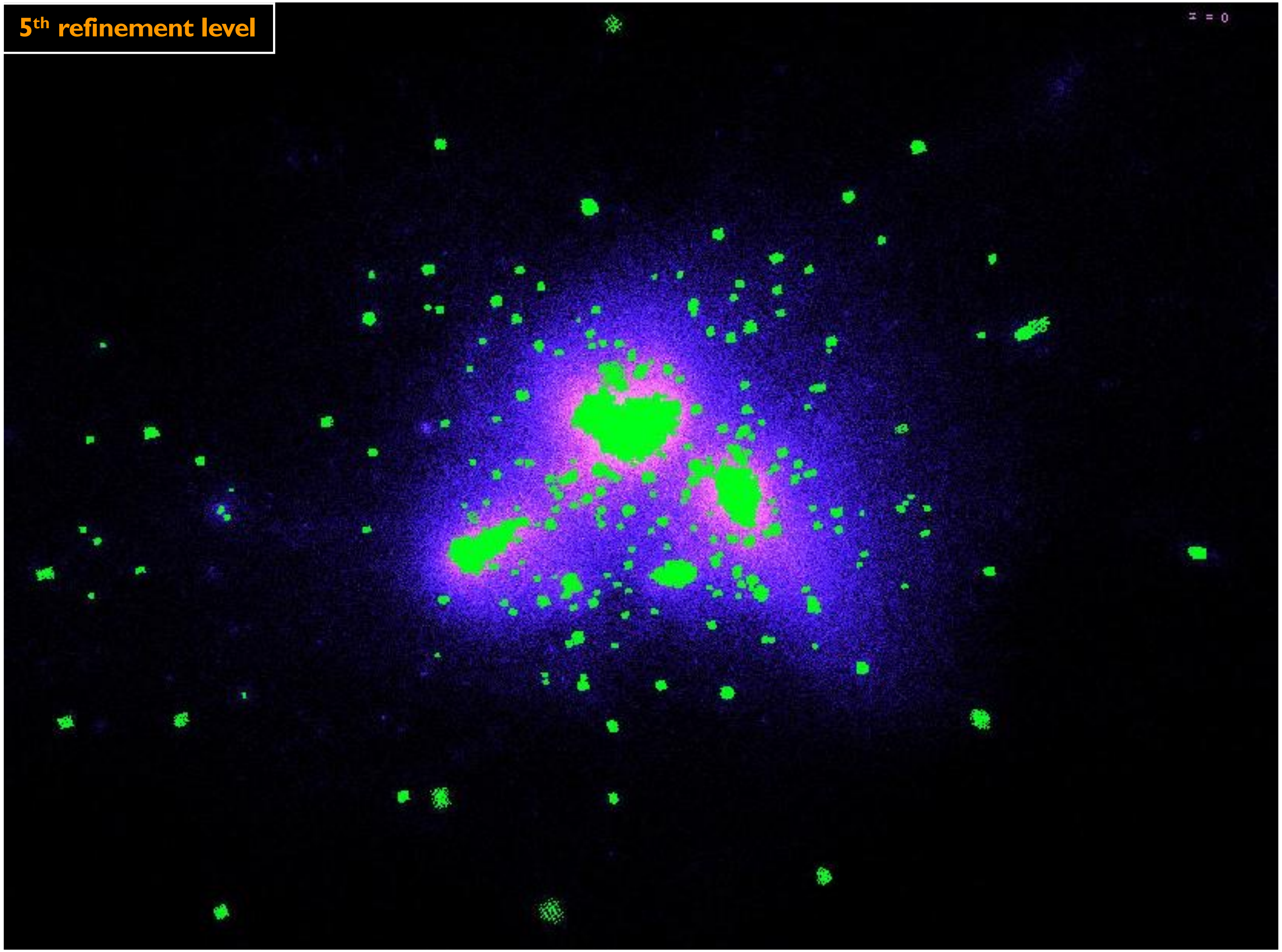

sample halo at z=0
z = 0
what about the adaptive meshes?

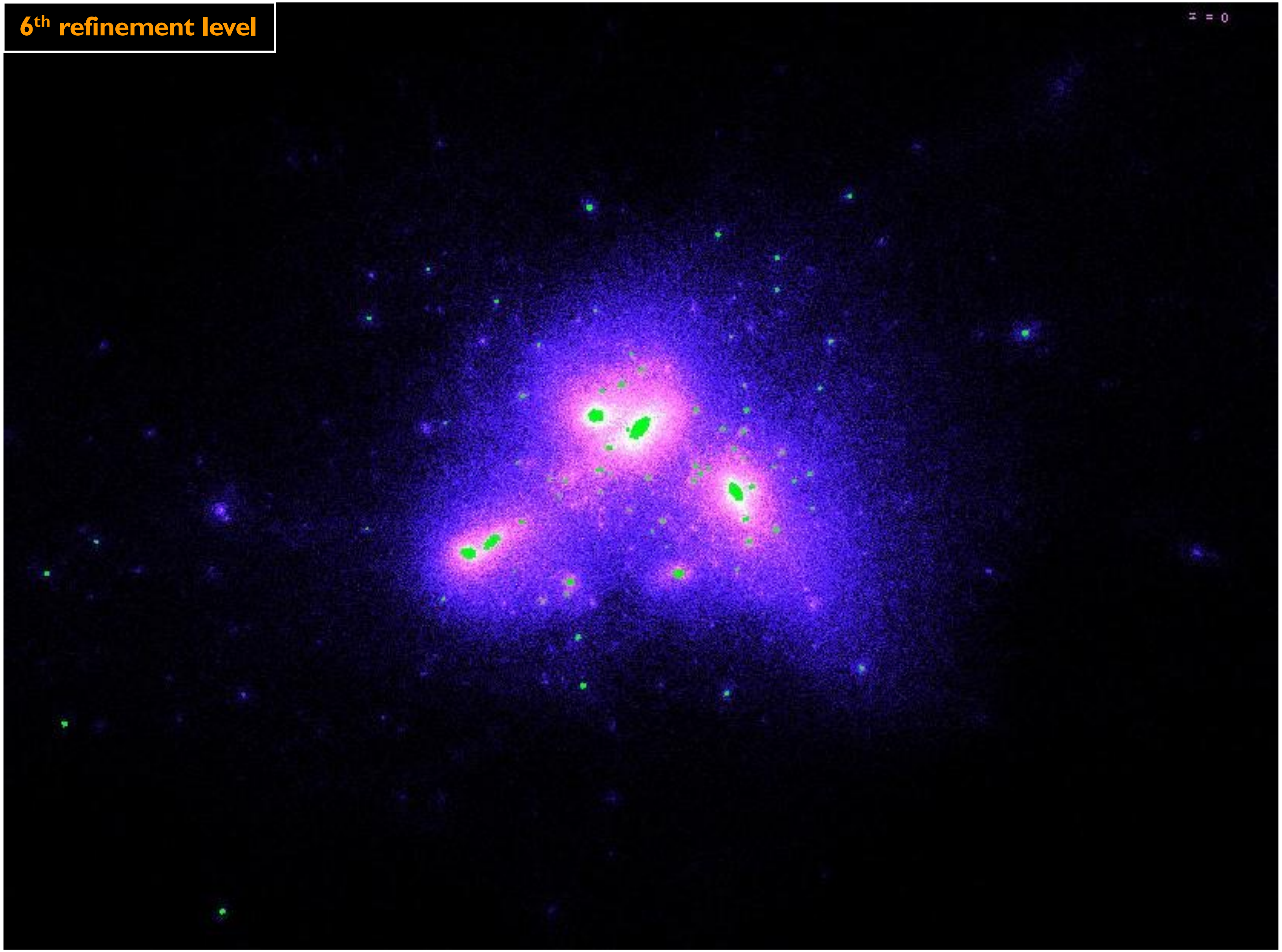

3rd refinement level

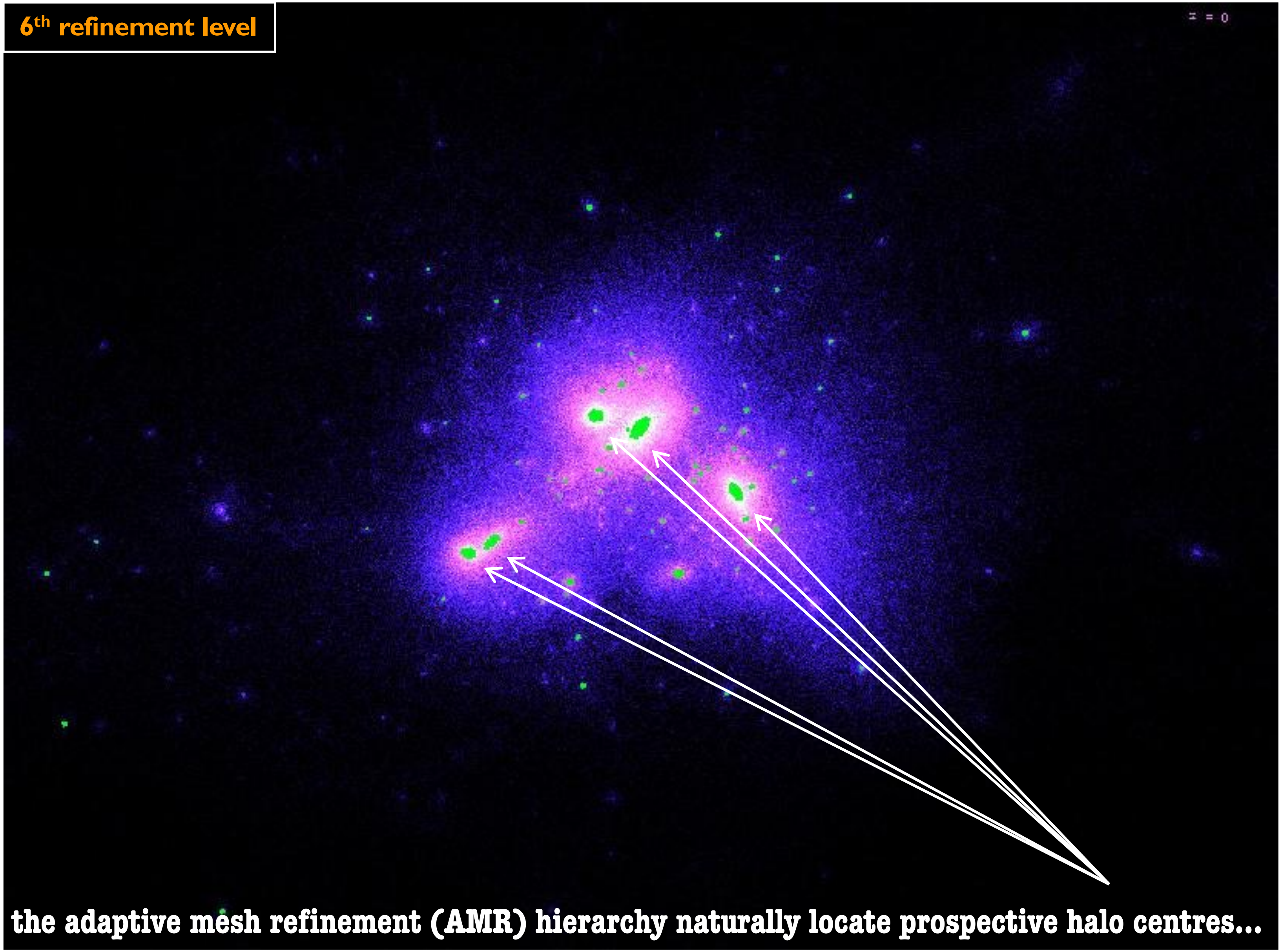
CONCEPT



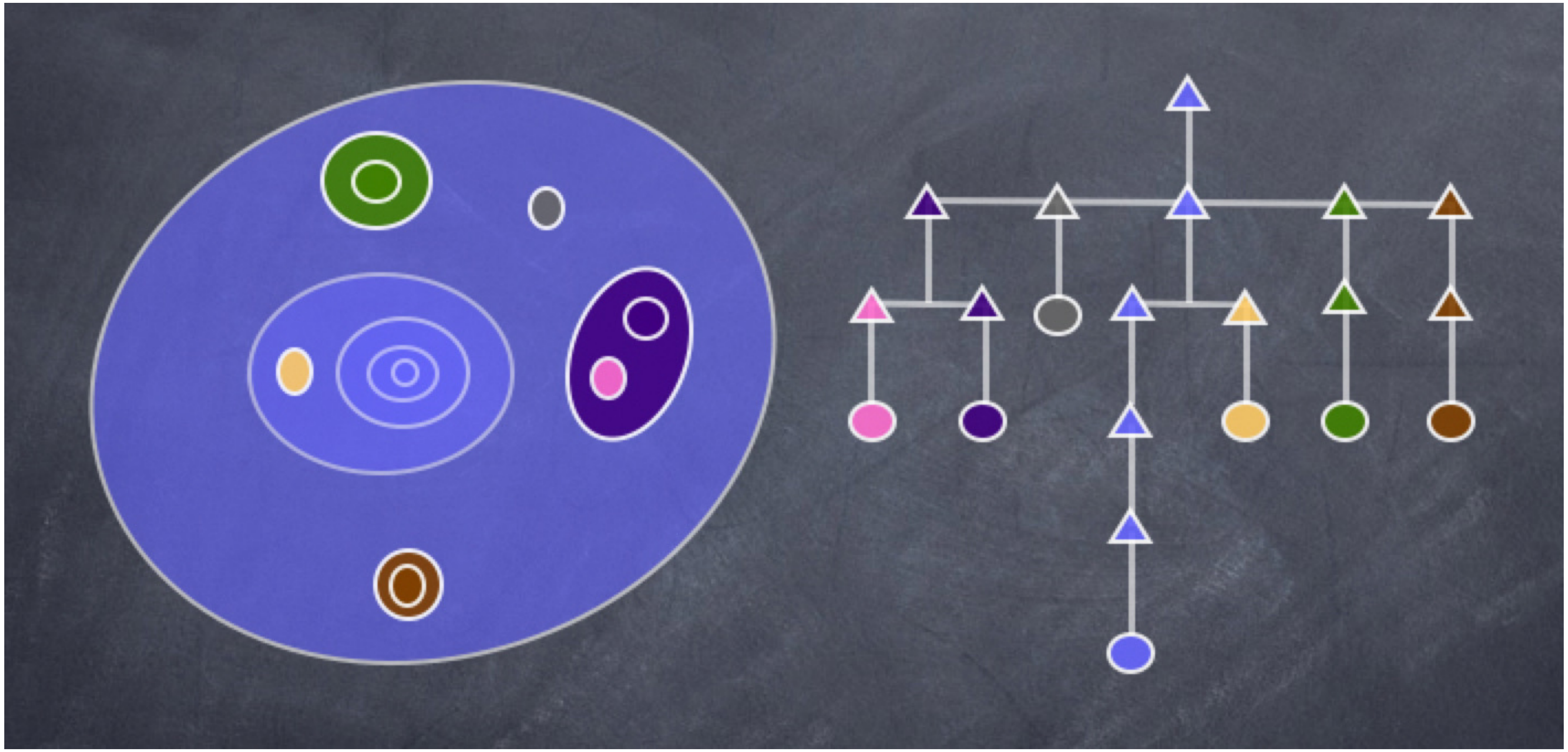
CONCEPT



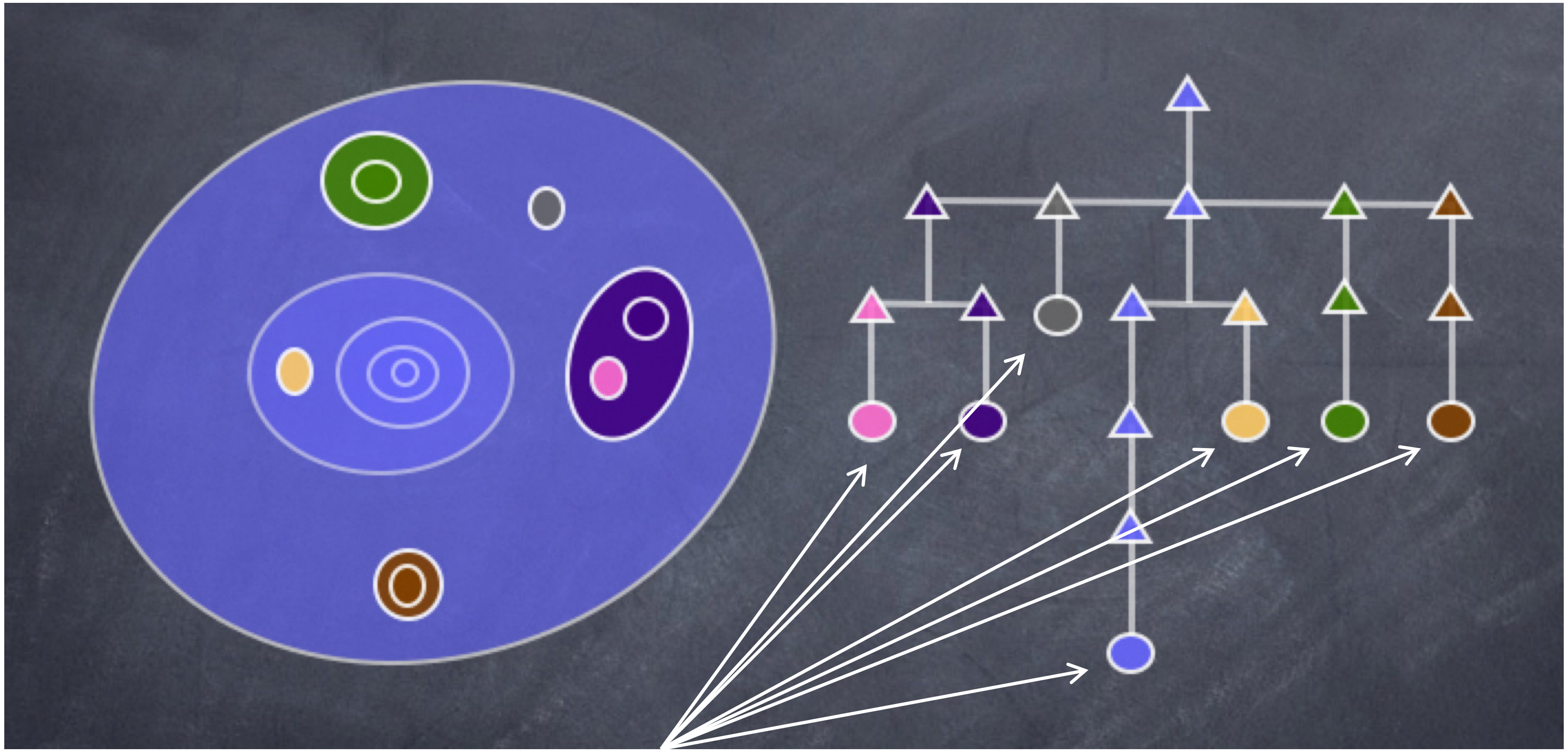

6th refinement level
z = 0


6th refinement level
z = 0
the adaptive mesh refinement (AMR) hierarchy naturally locate prospective halo centres...

- organize AMR hierarchy into a tree structure

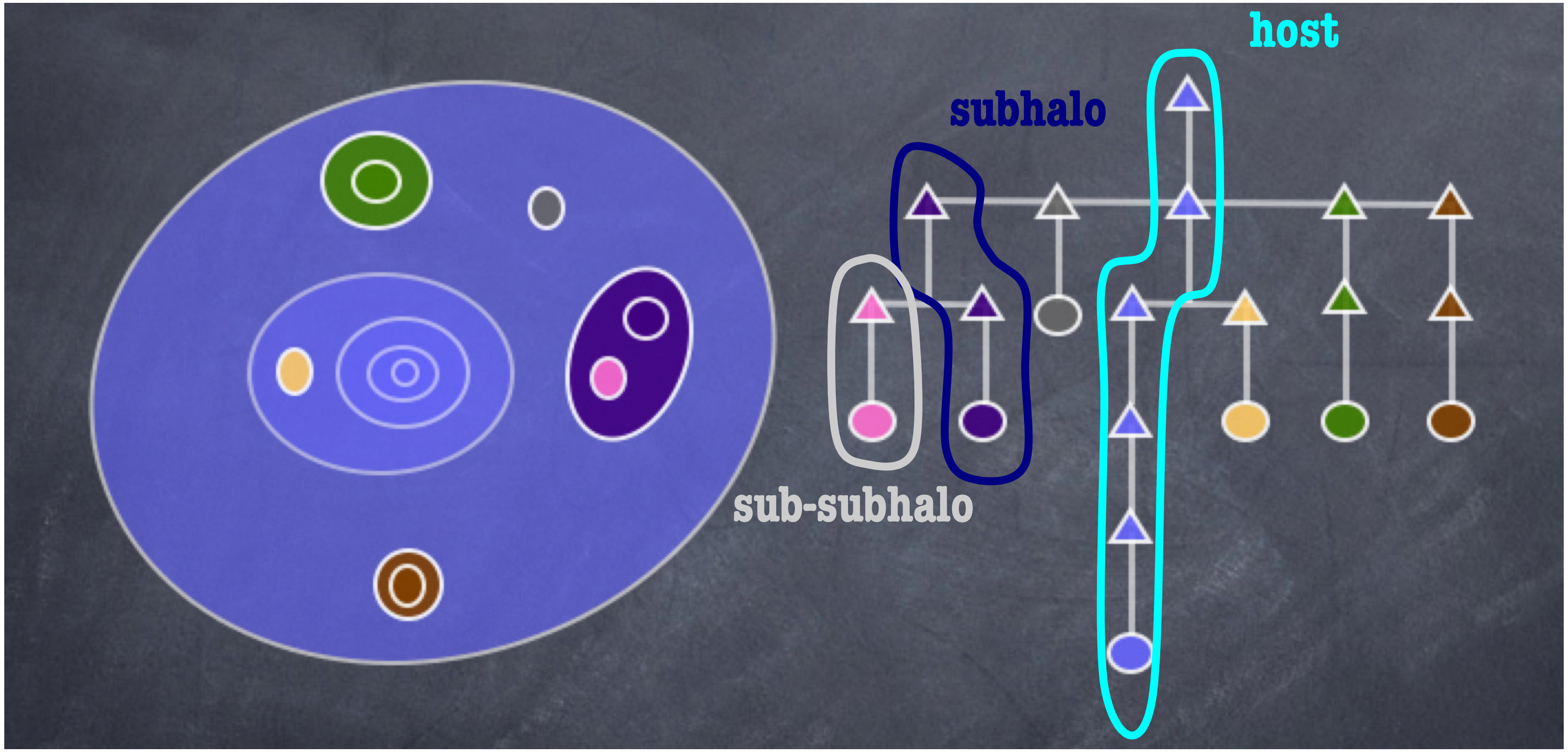


- organize AMR hierarchy into a tree structure



prospective halo centres...

- organize AMR hierarchy into a tree structure

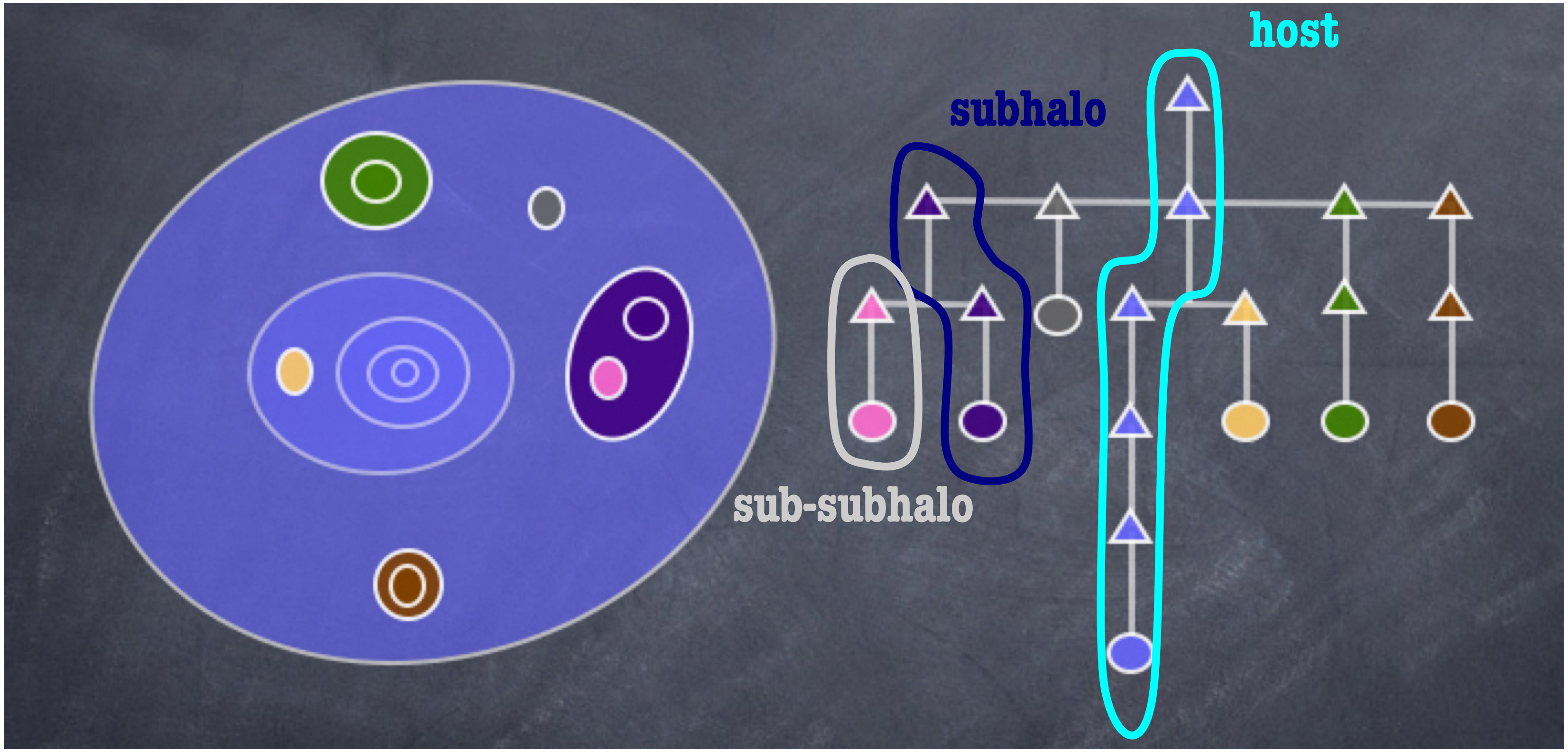


prospective halo centres...

...plus information about hosts, subhalos, sub-subhalos, etc.

- organize AMR hierarchy into a tree structure

The classification into host, subhalo, sub-subhalo, etc. will be explained later!

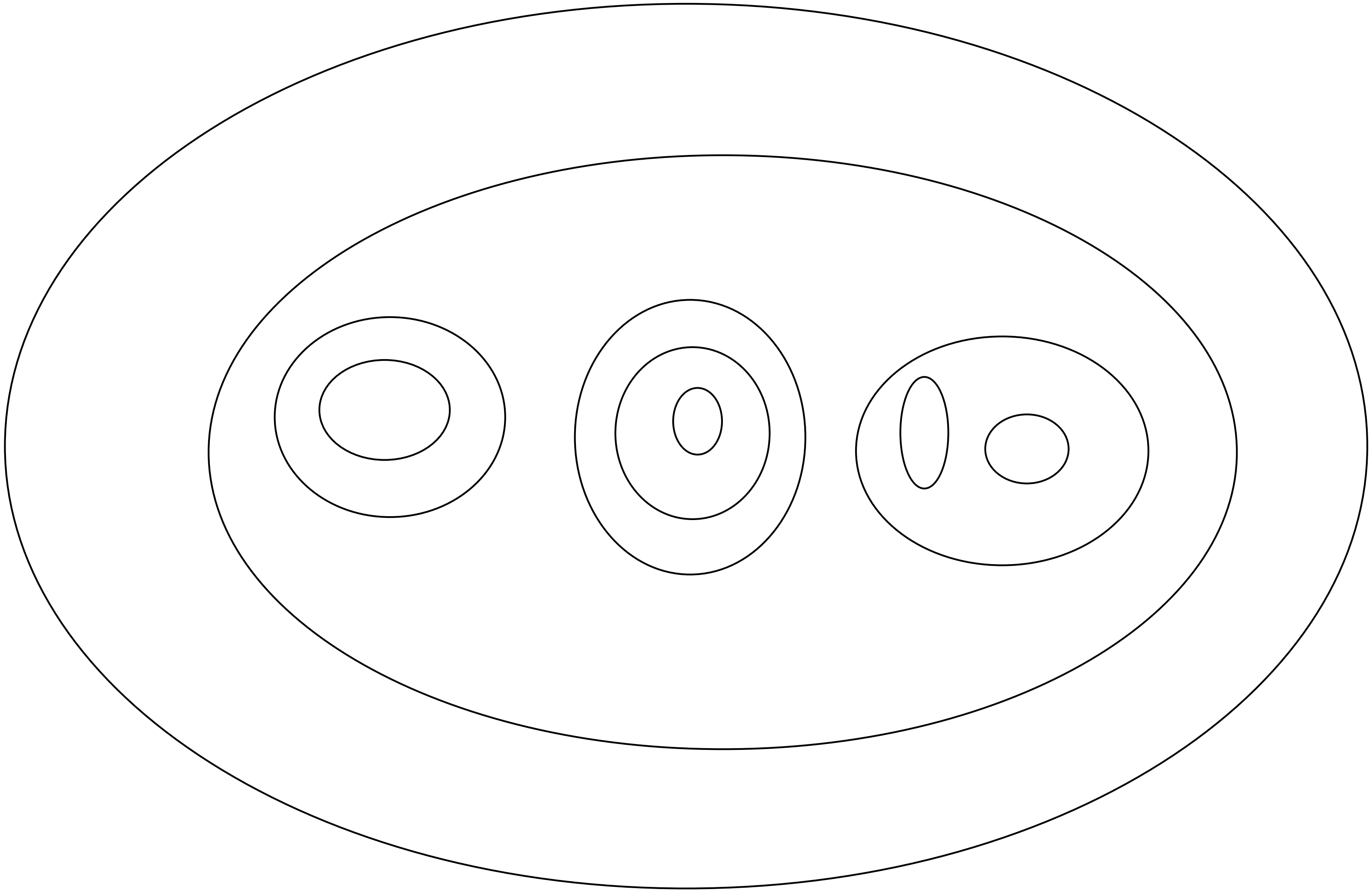


prospective halo centres...

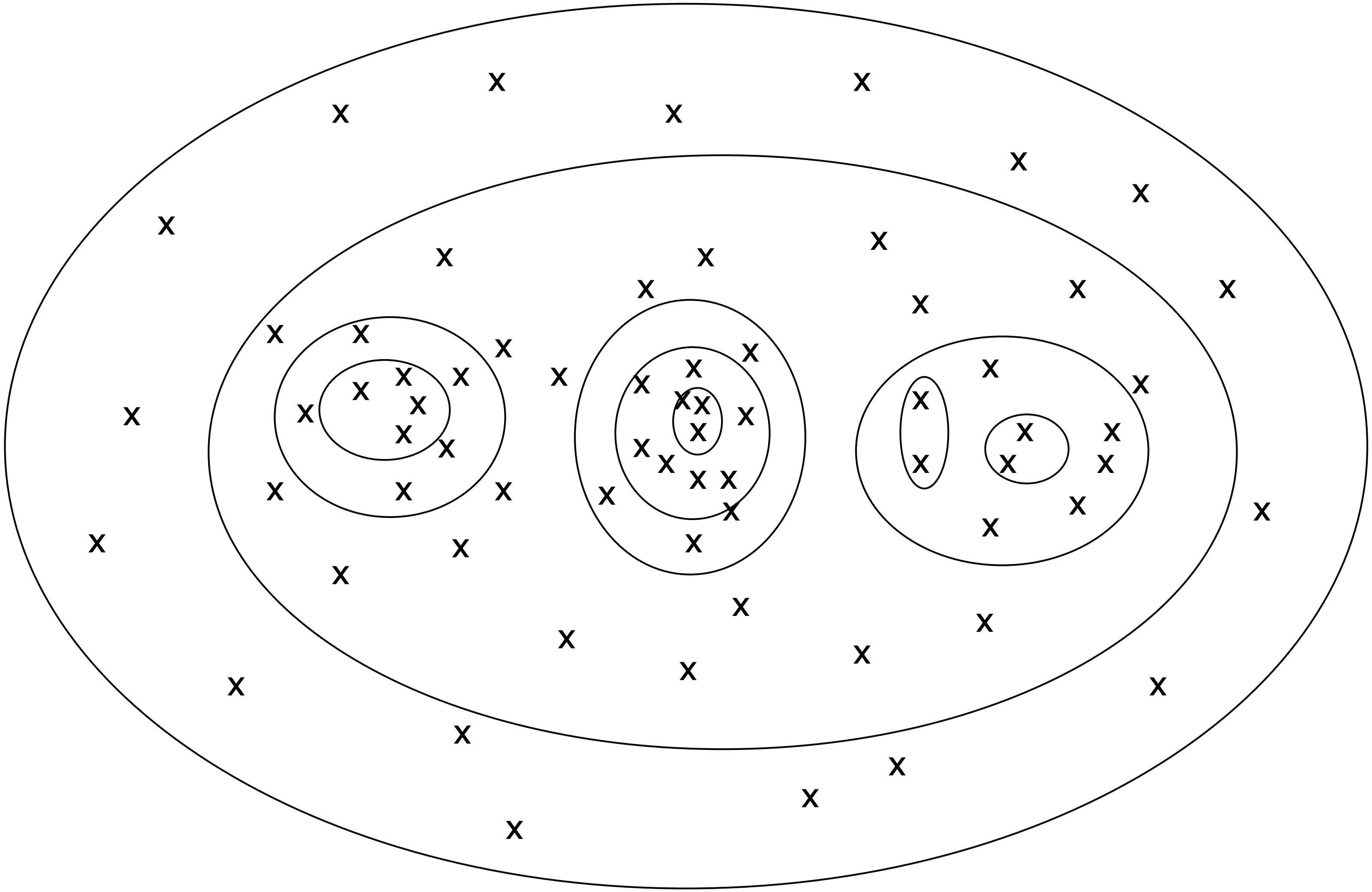
...plus information about hosts, subhalos, sub-subhalos, etc.

- AMR grids are isodensity contours

- AMR grids are isodensity contours

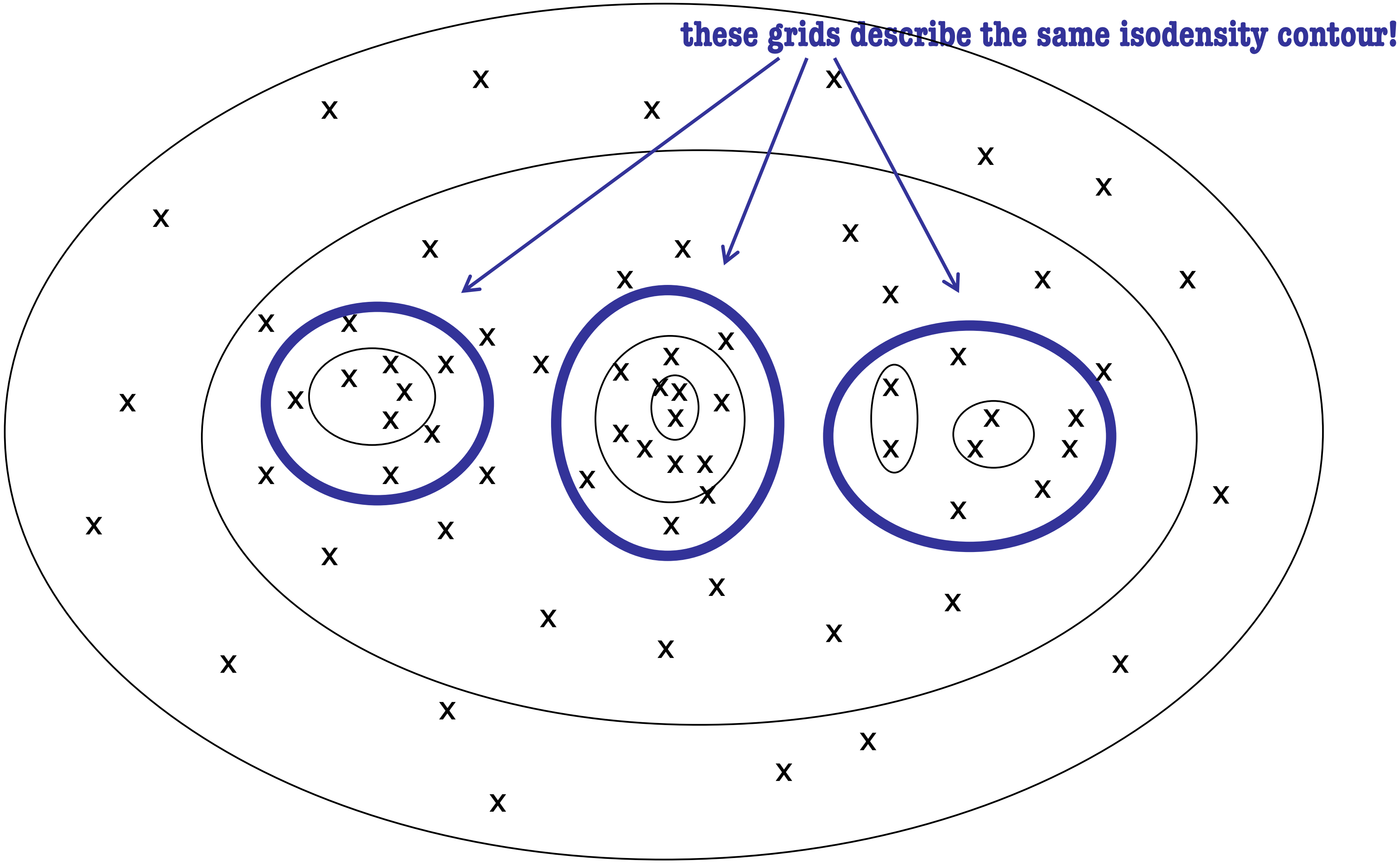


- AMR grids are isodensity contours...



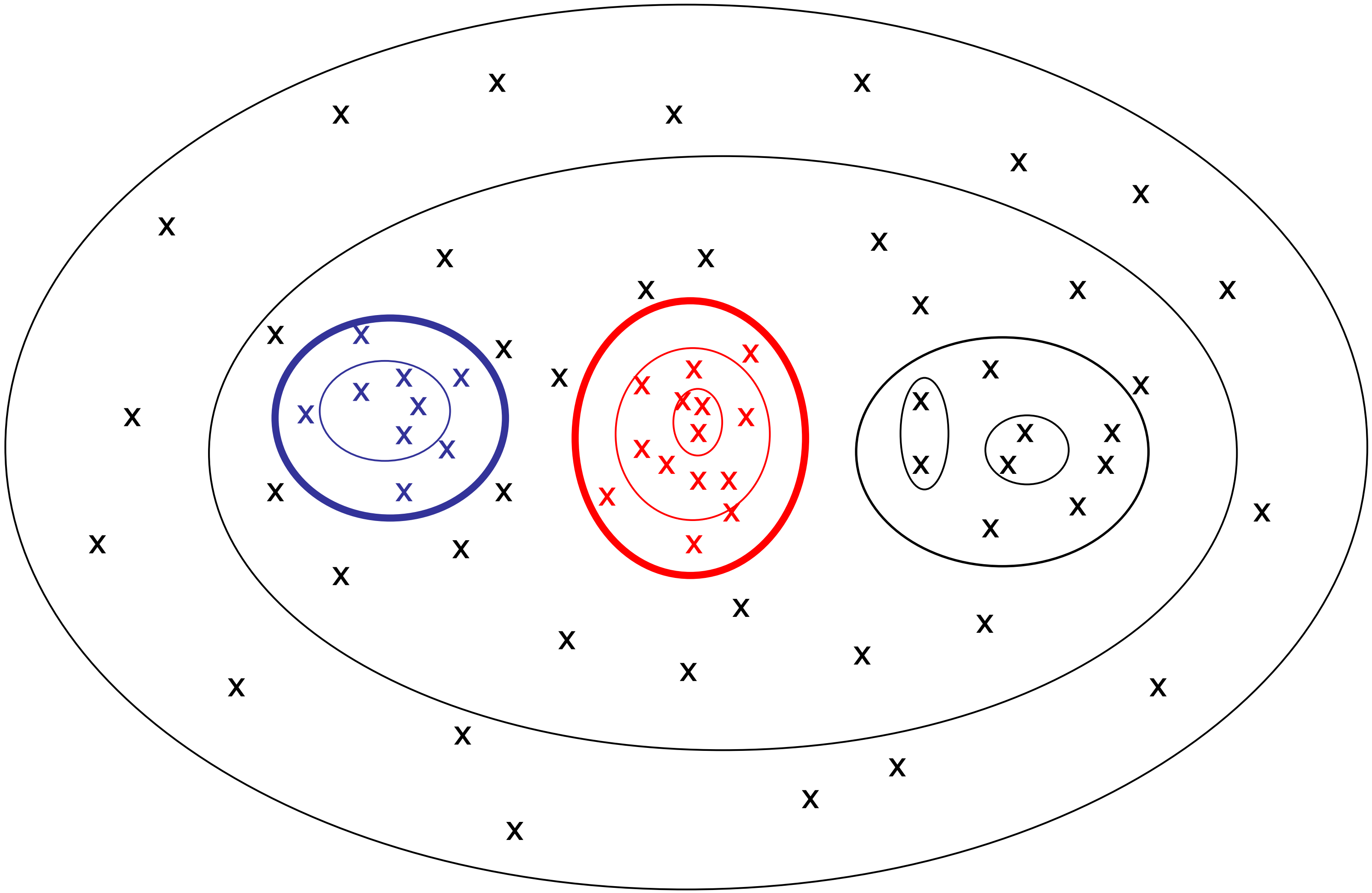
...encompassing particles!

- AMR grids are isodensity contours...

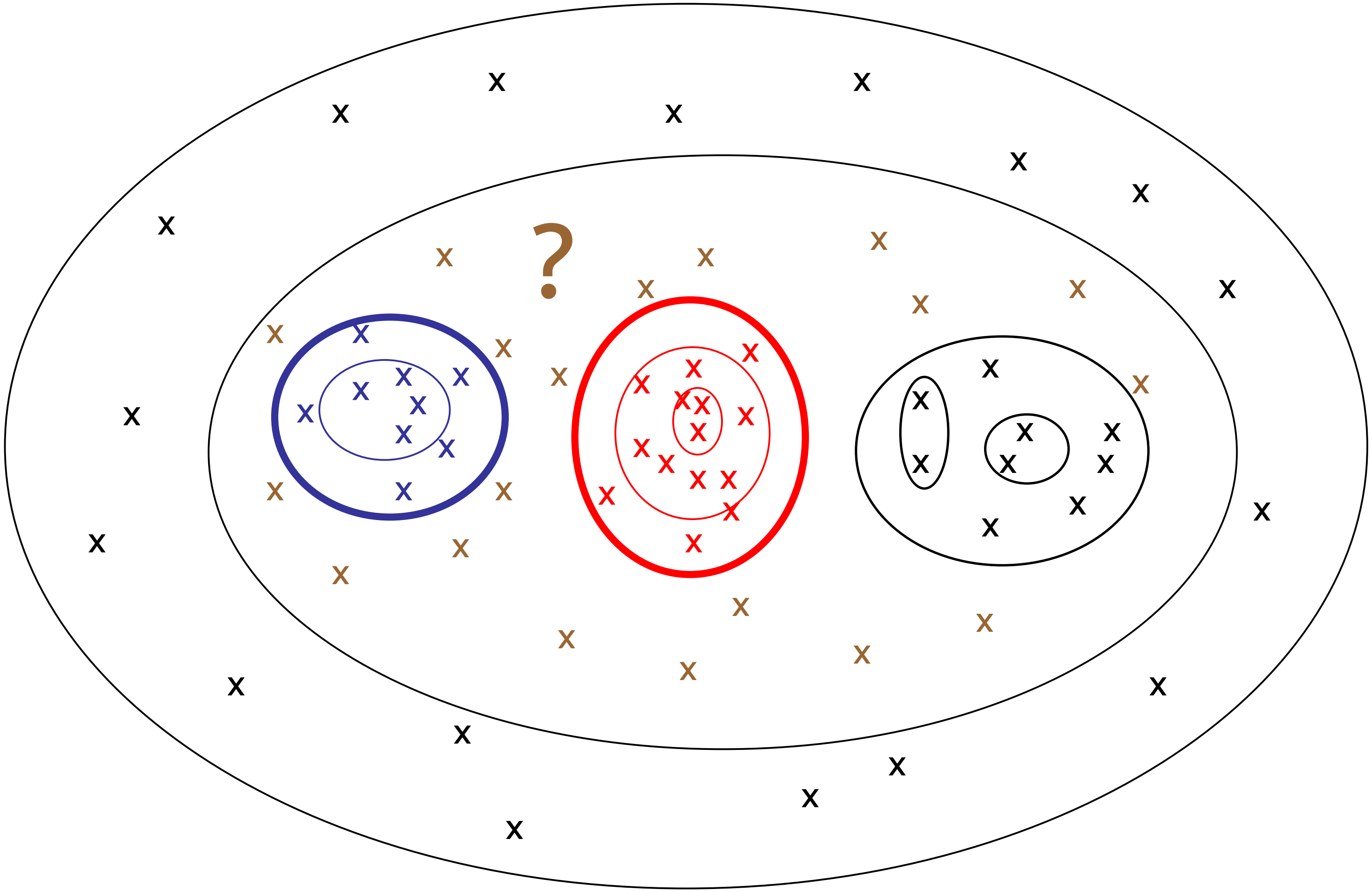


...encompassing particles!

1. collect all particles inside **unambiguous** isodensity contour

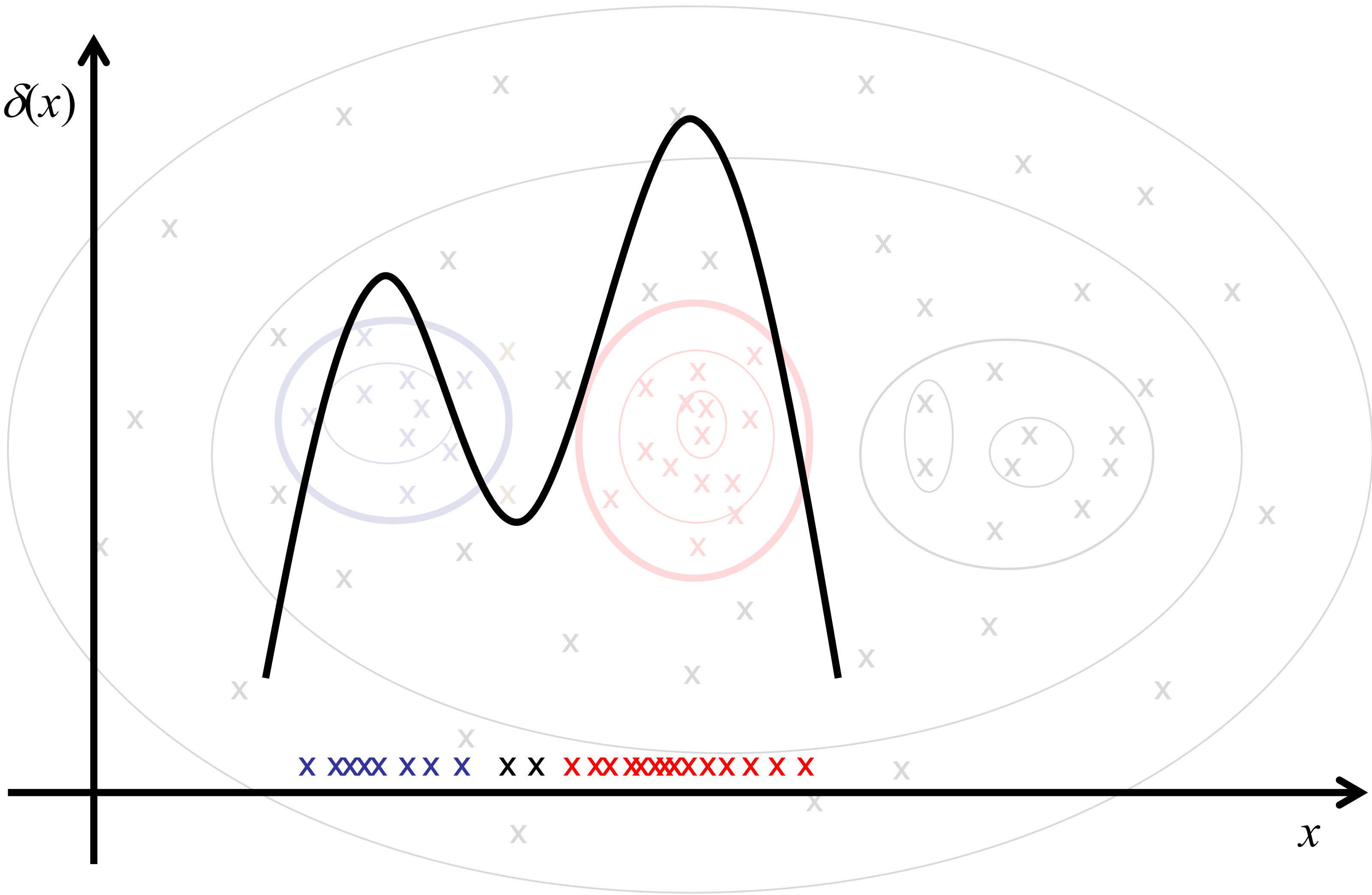


1. collect all particles inside **unambiguous** isodensity contour

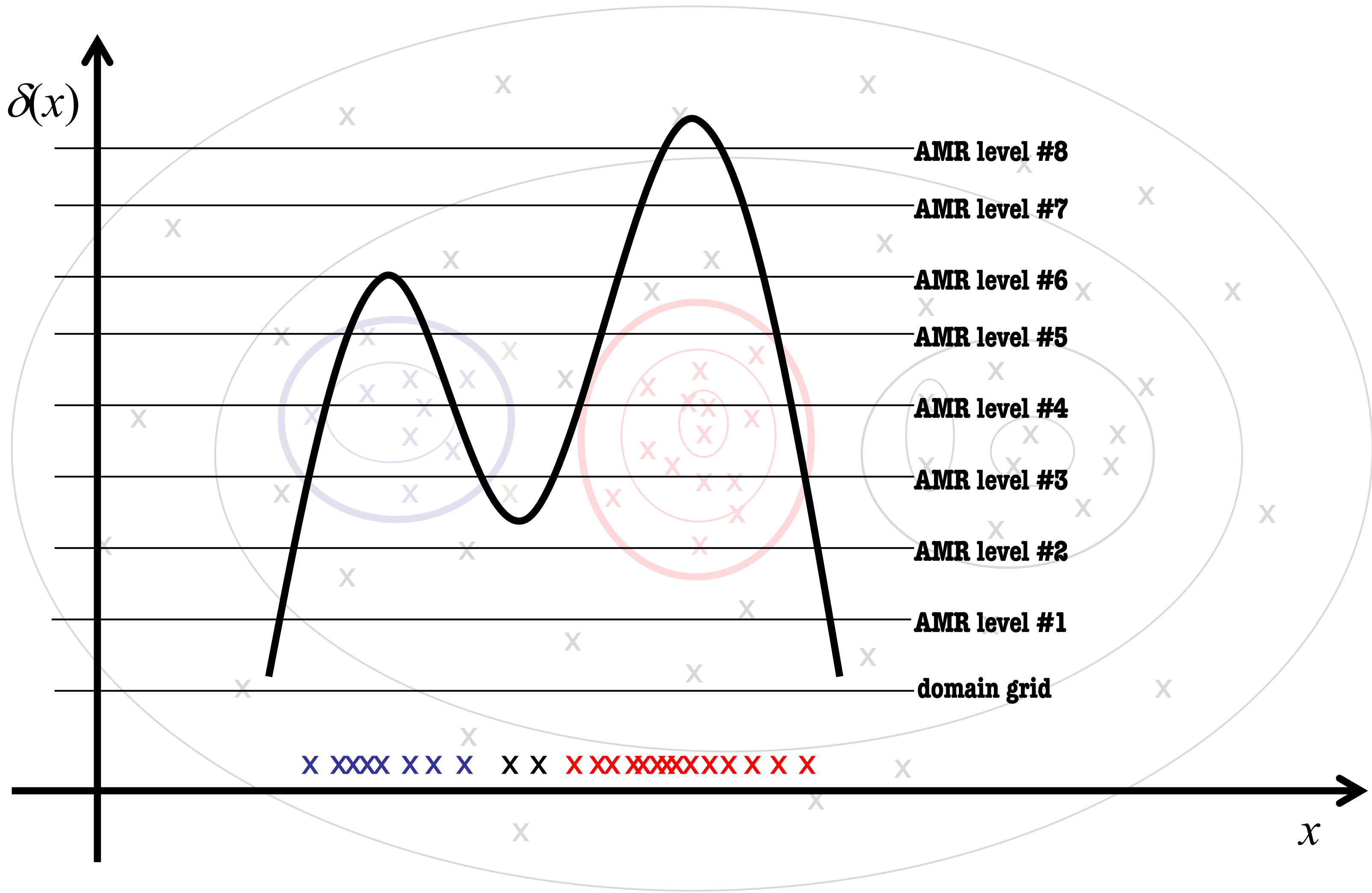


but what about **these** particles?

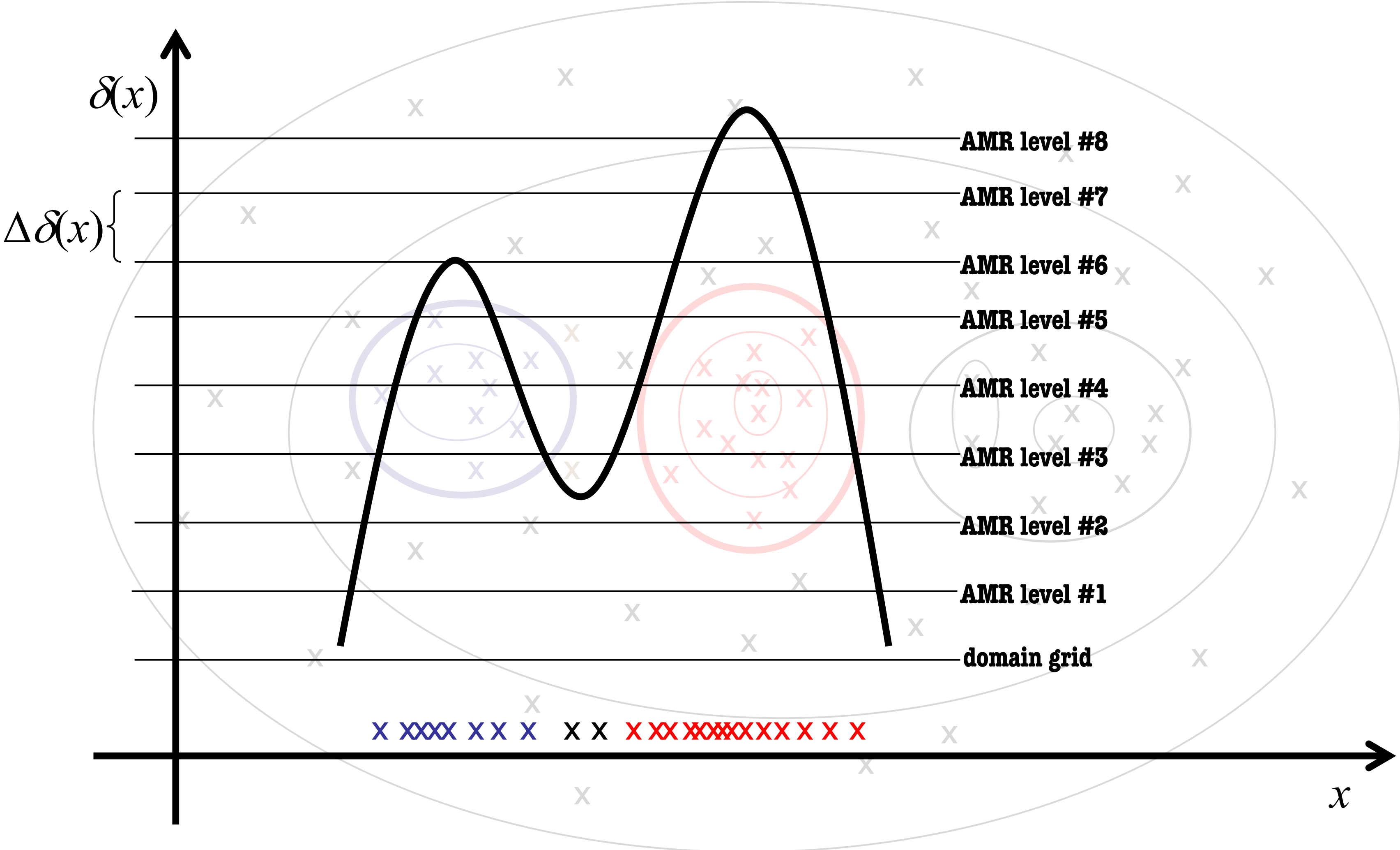
2. consider particles inside "half-distance-sphere", too!



2. consider particles inside "half-distance-sphere", too!

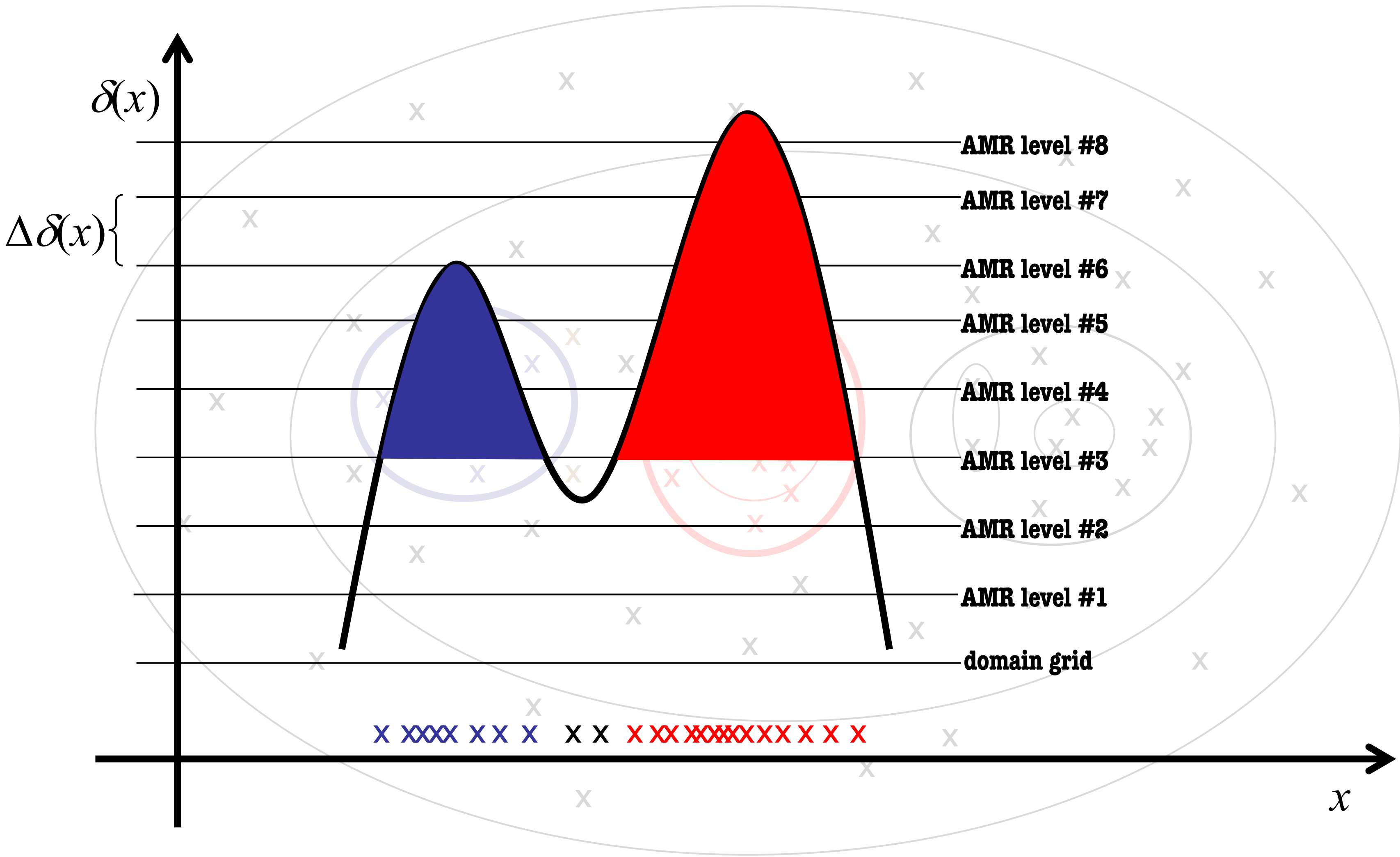


2. consider particles inside “half-distance-sphere”, too!



Note: $\Delta\delta(x)$ is determined by the refinement criterion...

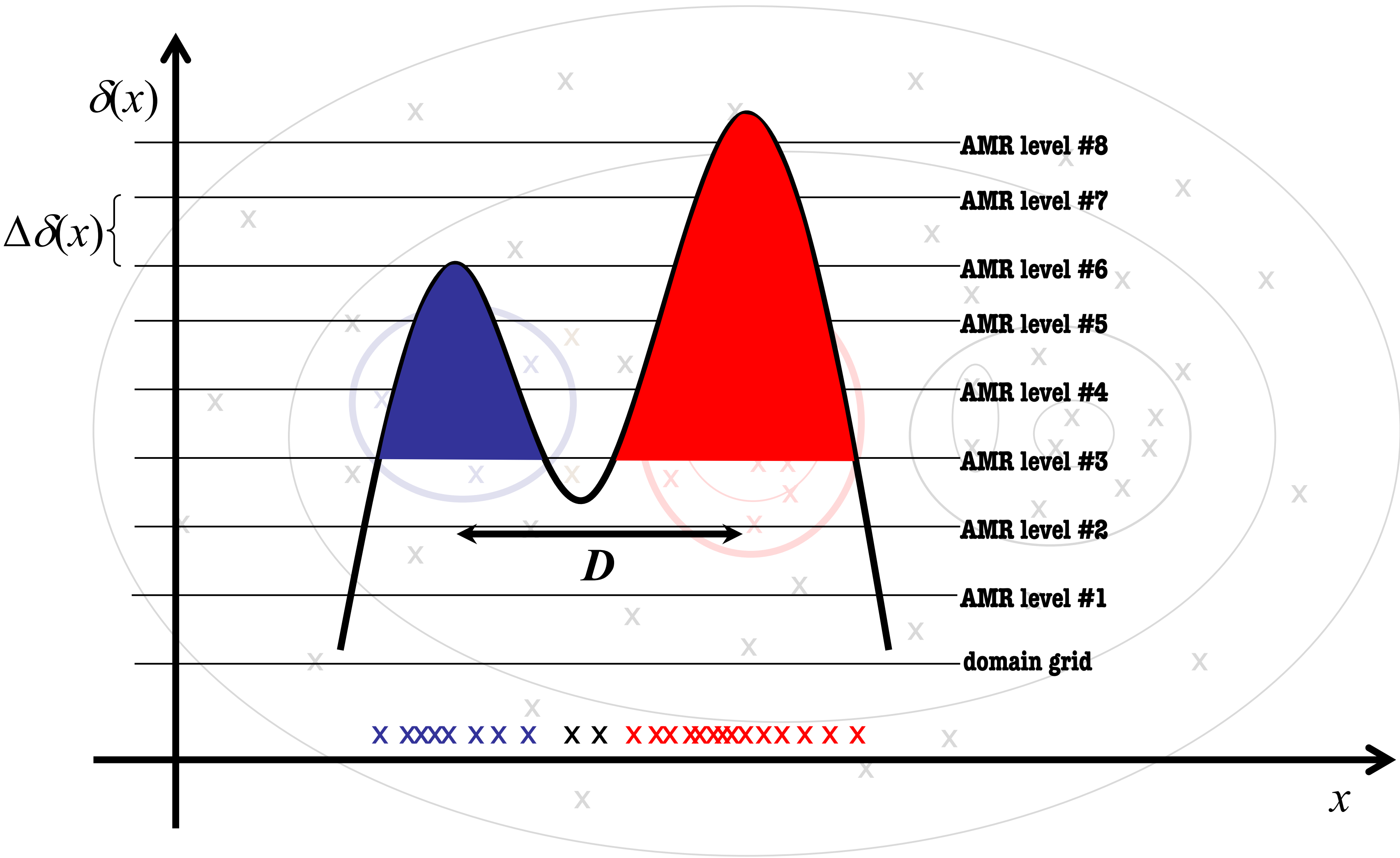
2. consider particles inside "half-distance-sphere", too!



Note: $\Delta\delta(x)$ is determined by the refinement criterion...

AHF - AMIGA's HALO FINDER

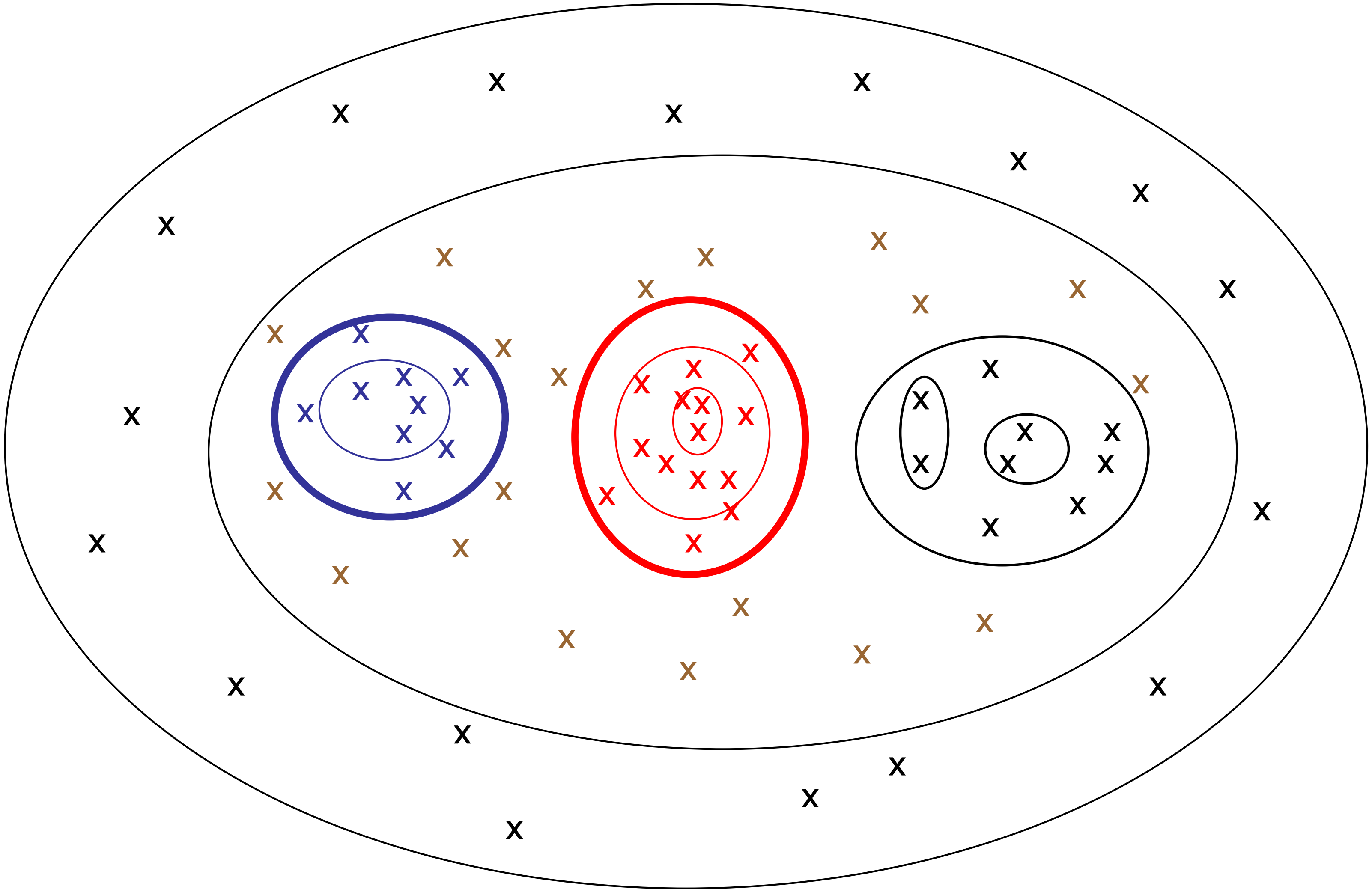
2. consider particles inside "half-distance-sphere", too!



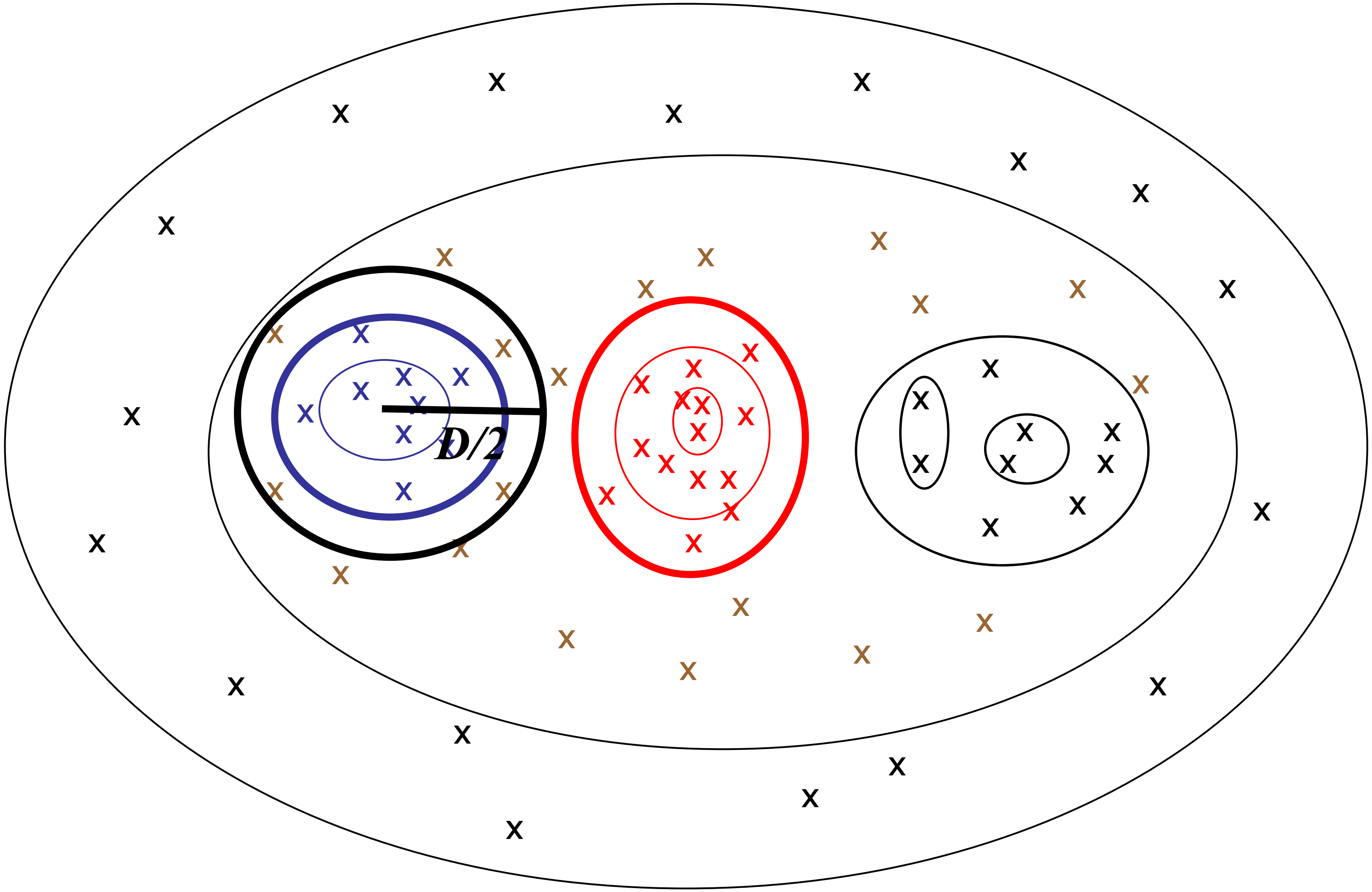
Note: $\Delta\delta(x)$ is determined by the refinement criterion...

AHF - Amiga's Halo Finder

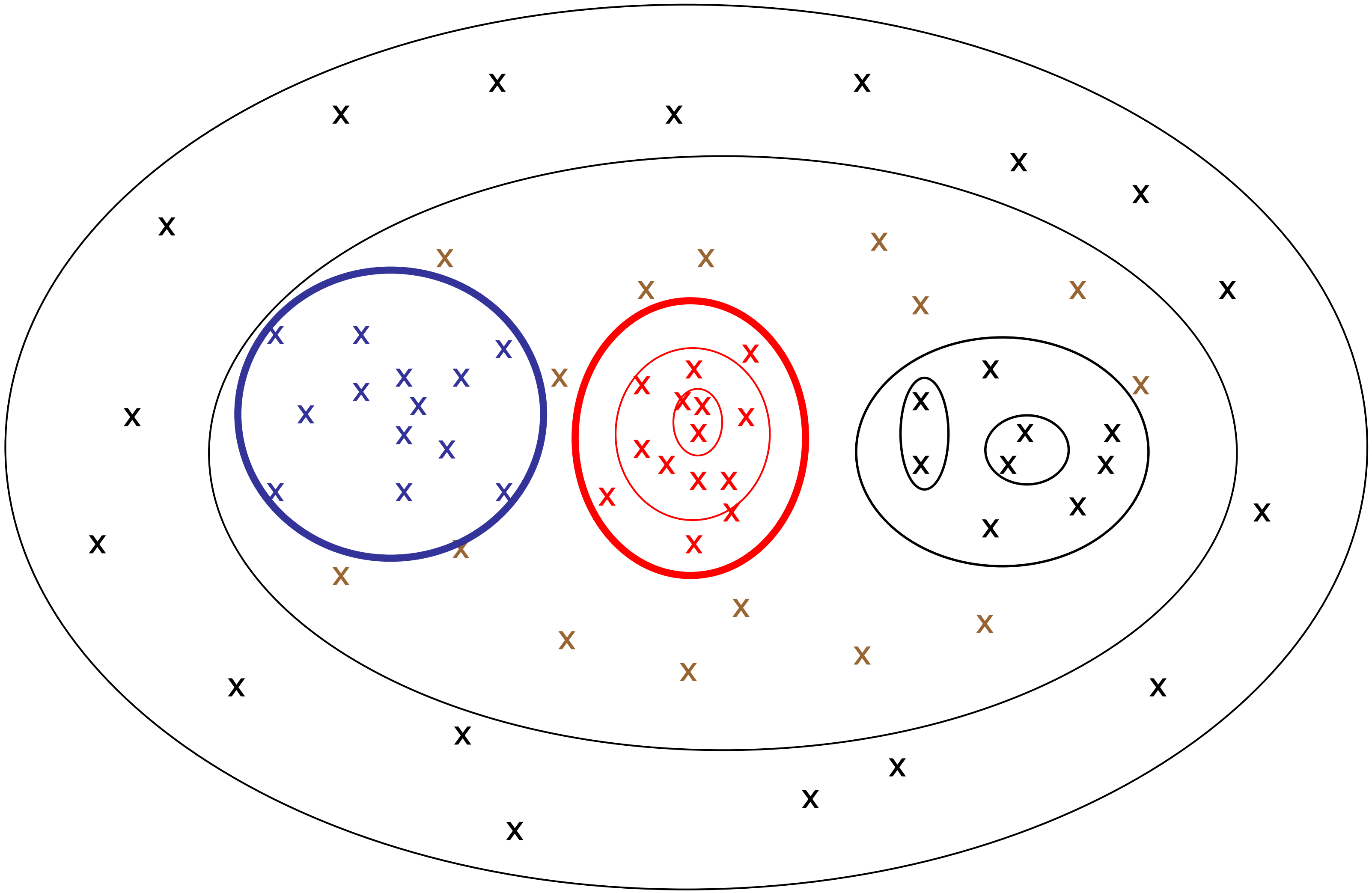
2. consider particles inside "half-distance-sphere", too!



2. consider particles inside "half-distance-sphere", too!

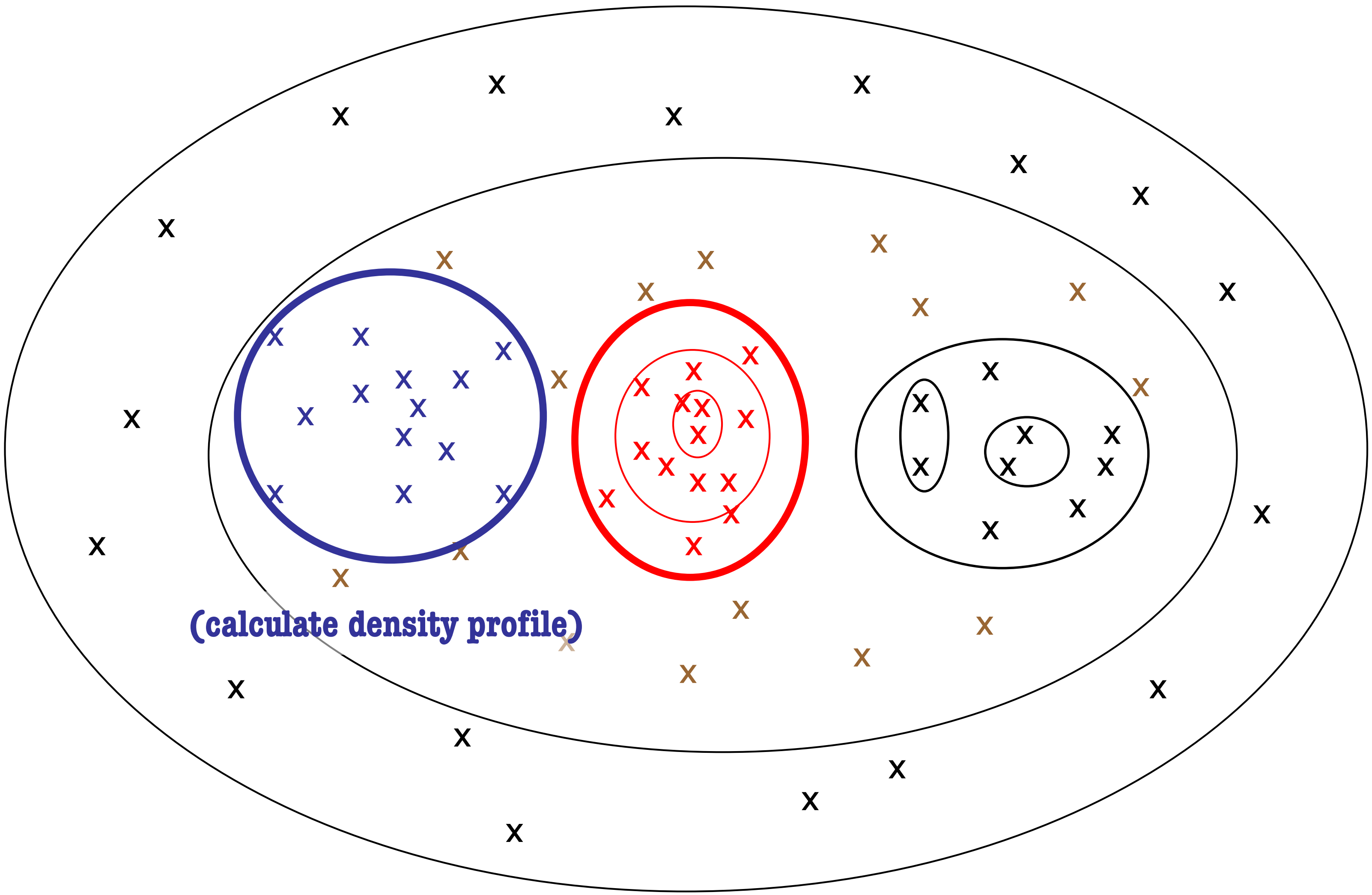


2. consider particles inside "half-distance-sphere", too!



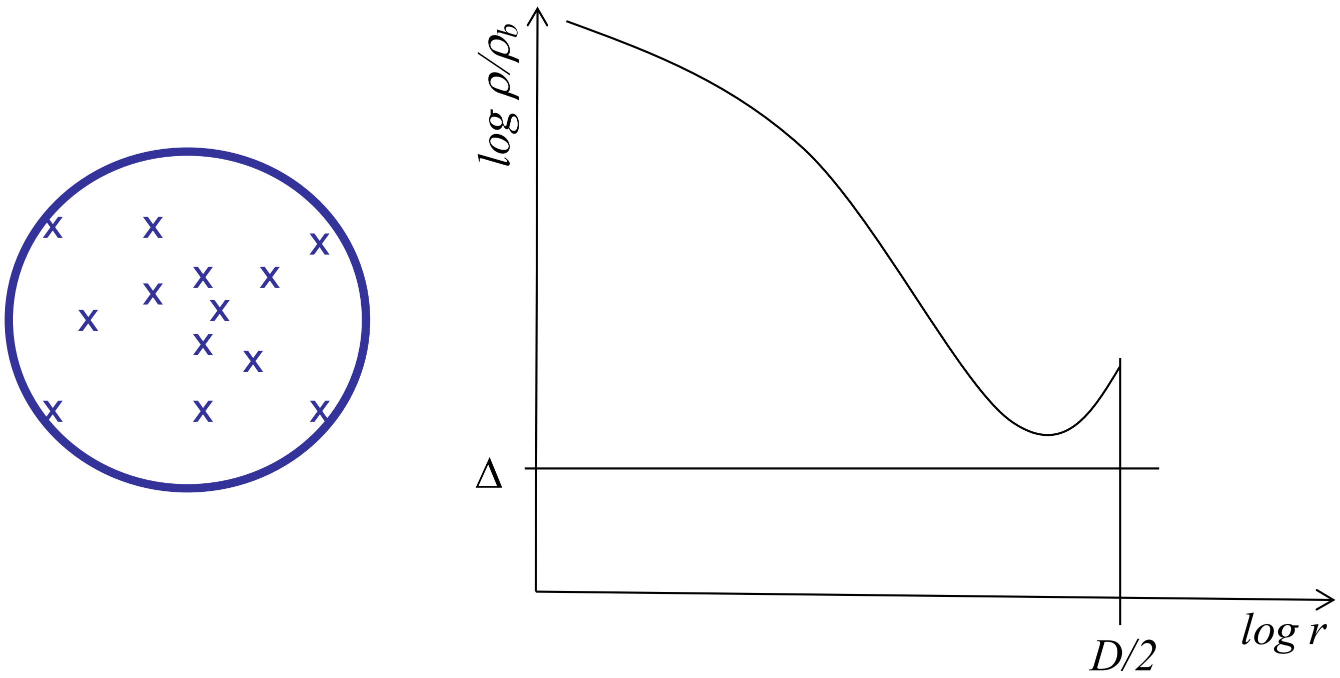
Note: particles not bound to the one **halo**, will be considered for boundness by the other **halo**

2. consider particles inside "half-distance-sphere", too!



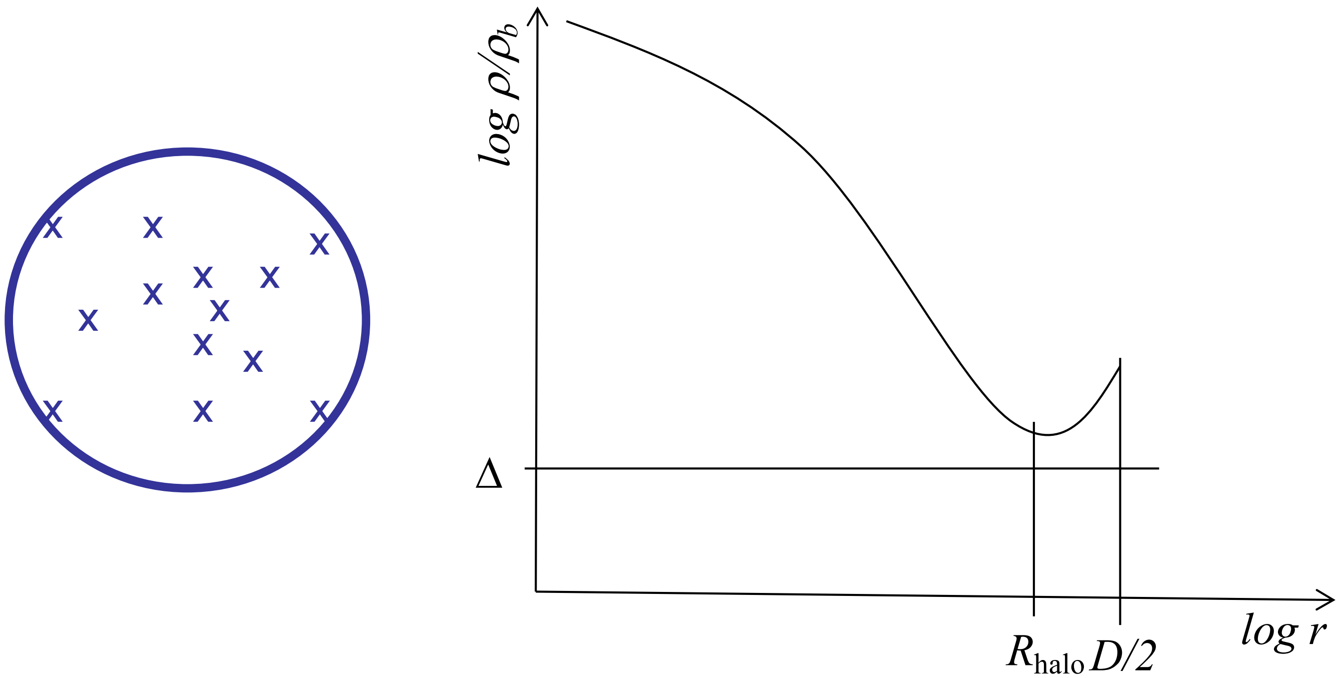
which halo comes first relates to the classification into host, subhalo, sub-subhalo, etc...

3. determine halo edge (prior to unbinding)



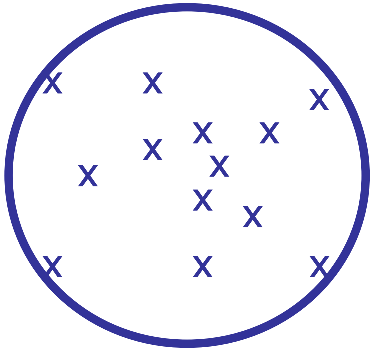
Δ is calculated inside ***AHF*** and depends on cosmology and redshift!

3. determine halo edge (prior to unbinding)

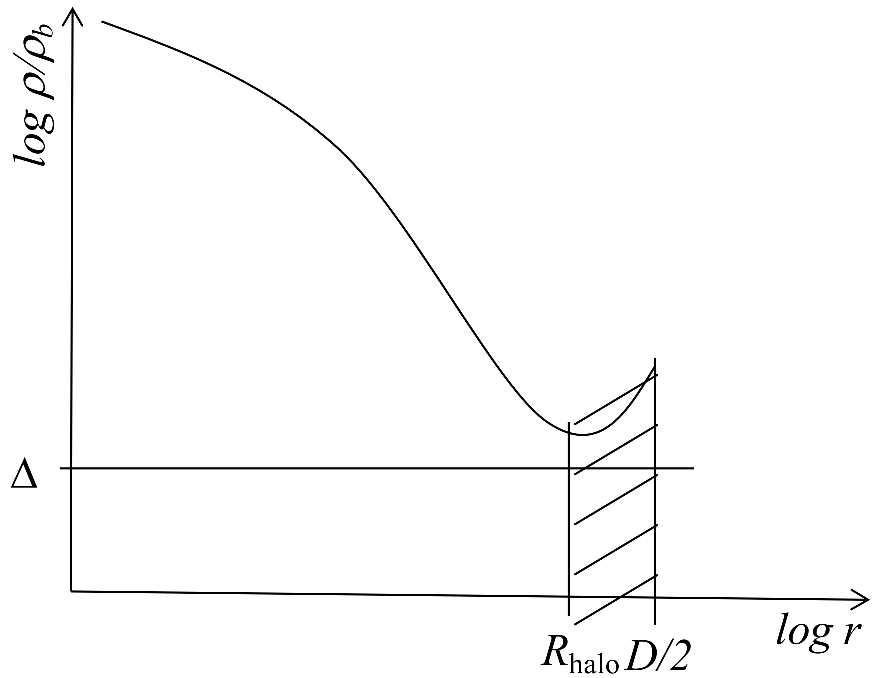


Δ is calculated inside ***AHF*** and depends on cosmology and redshift!

3. determine halo edge (prior to unbinding)

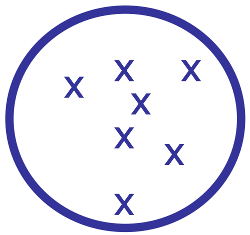


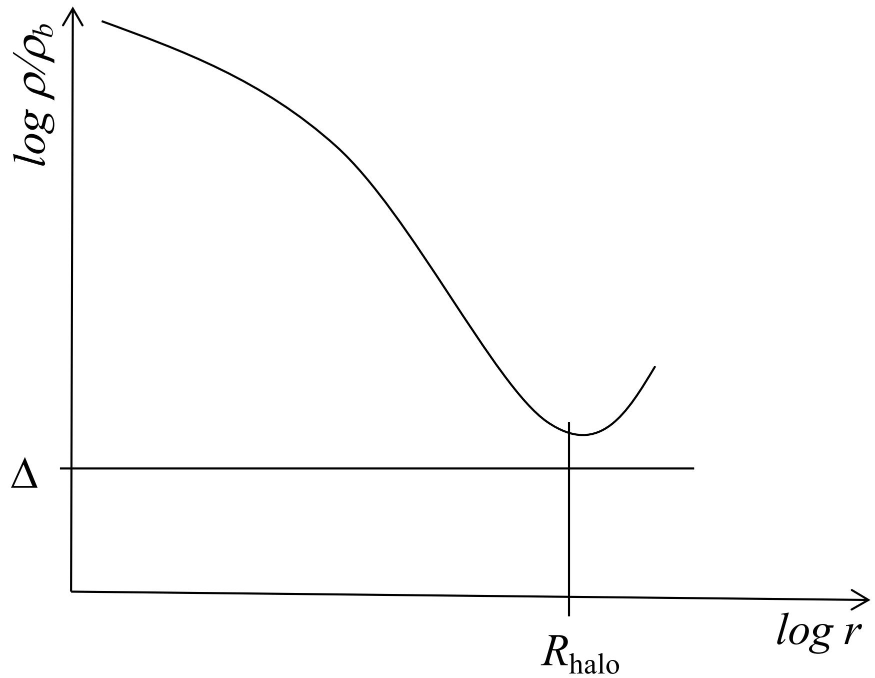
(remove outliers)



Δ is calculated inside ***AHF*** and depends on cosmology and redshift!

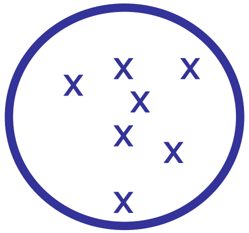
3. determine halo edge (prior to unbinding)





Δ is calculated inside ***AHF*** and depends on cosmology and redshift!

4. iteratively remove unbound particles

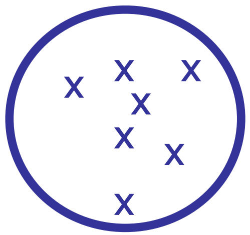


(remove unbound particles)

4. iteratively remove unbound particles

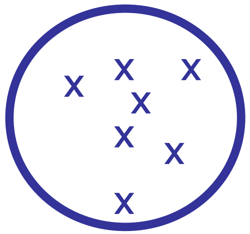
assume spherical symmetry:

$$\rho = \rho(r)$$



(remove unbound particles)

4. iteratively remove unbound particles



(remove unbound particles)

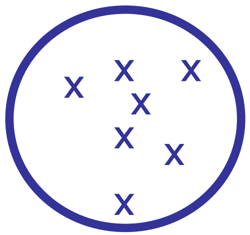
assume spherical symmetry:

$$\rho = \rho(r)$$

solve Poisson's equation:

$$\Delta\varphi = 4\pi G\rho$$

4. iteratively remove unbound particles



(remove unbound particles)

assume spherical symmetry:

$$\rho = \rho(r)$$

solve Poisson's equation:

$$\Delta\varphi = 4\pi G\rho$$

first integration...

$$\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d\varphi}{dr}\right) = 4\pi G\rho$$

$$\psi = r^2\frac{d\varphi}{dr}$$

$$\frac{1}{r^2}\frac{d}{dr}(\psi) = 4\pi G\rho$$

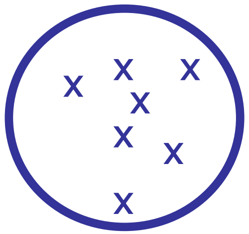
$$\frac{d\psi}{dr} = 4\pi G\rho r^2$$

$$\psi(r) - \psi(0) = 4\pi G\int_0^r \rho r'^2 dr'$$

$$\psi(0) = \left[r^2\frac{d\varphi}{dr}\right]_{r=0} = 0$$

$$\psi(r) = GM(<r)$$

4. iteratively remove unbound particles



(remove unbound particles)

assume spherical symmetry:

$$\rho = \rho(r)$$

solve Newton's force law:

$$\frac{d\varphi}{dr} = \frac{GM(<r)}{r^2}$$

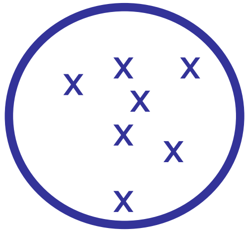
second integration...

$$\varphi(r) = G\int_0^r \frac{M(<r')}{r'^2}\,dr' + \varphi(0)$$

unbound, if...

$$v > v_{\mathrm{esc}} = \sqrt{2|\varphi|}$$

4. iteratively remove unbound particles



(remove unbound particles)

assume spherical symmetry:

$$\rho = \rho(r)$$

solve Newton's force law:

$$\frac{d\varphi}{dr} = \frac{GM(<r)}{r^2}$$

second integration...

$$\varphi(r) = G\int_0^r \frac{M(<r')}{r'^2}\,dr' + \varphi(0) \quad ?$$

unbound, if...

$$v > v_{esc} = \sqrt{2|\varphi|}$$

4. iteratively remove unbound particles

potential normalisation:

$$\varphi(\infty) = G\int_0^\infty \frac{M(<r')}{r'^2}dr' + \varphi(0)$$

$$= G\int_0^{R_{vir}} \frac{M(<r')}{r'^2}dr' + G\int_{R_{vir}}^\infty \frac{M(<r')}{r'^2}dr' + \varphi(0)$$

$$= G\int_0^{R_{vir}} \frac{M(<r')}{r'^2}dr' + GM_{vir}\int_{R_{vir}}^\infty \frac{1}{r'^2}dr' + \varphi(0)$$

$$= G\int_0^{R_{vir}} \frac{M(<r')}{r'^2}dr' + GM_{vir}\left[-\frac{1}{r}\right]_{R_{vir}}^\infty + \varphi(0)$$

$$= G\int_0^{R_{vir}} \frac{M(<r')}{r'^2}dr' + G\frac{M_{vir}}{R_{vir}} + \varphi(0)$$

assume spherical symmetry:

$$\rho = \rho(r)$$

solve Newton's force law:

$$\frac{d\varphi}{dr} = \frac{GM(<r)}{r^2}$$

second integration...

$$\varphi(r) = G\int_0^r \frac{M(<r')}{r'^2}dr' + \varphi(0)$$?

unbound, if...

$$v > v_{esc} = \sqrt{2|\varphi|}$$

4. iteratively remove unbound particles

$$\varphi(r) = G\int_0^r \frac{M(<r')}{r'^2}dr' - \varphi_0$$

with: $$\varphi_0 = G\left(\frac{M_{vir}}{R_{vir}} + \int_0^{R_{vir}} \frac{M(<r')}{r'^2}dr'\right)$$

the integrals can be readily evaluated in cosmological simulations...

4. iterative remove unbound particles

$$\varphi(r) = G\int_0^r \frac{M(<r')}{r'^2}dr' - \varphi_0$$

order particles with respect to distance:

$$\int_0^r \frac{M(<r')}{r'^2}dr' = \int_0^{r_1} \frac{M(<r)}{r^2}dr + \int_{r_1}^{r_2} \frac{M(<r)}{r^2}dr + \ldots + \int_{r_{N-1}}^{r_N} \frac{M(<r)}{r^2}dr$$

$$= \frac{m_1}{r_1^2}r_1 + \frac{m_1+m_2}{r_2^2}\left|r_2 - r_1\right| + \frac{m_1+m_2+m_3}{r_3^2}\left|r_3 - r_2\right| + \ldots$$

4. **iteratively** remove unbound particles

$$\varphi(r) = G\int_0^r \frac{M(<r')}{r'^2}dr' - \varphi_0$$

order particles with respect to distance:

$$\int_0^r \frac{M(<r')}{r'^2}dr' = \int_0^{r_1} \frac{M(<r)}{r^2}dr + \int_{r_1}^{r_2} \frac{M(<r)}{r^2}dr + \ldots + \int_{r_{N-1}}^{r_N} \frac{M(<r)}{r^2}dr$$

$$= \frac{m_1}{r_1^2}r_1 + \frac{m_1+m_2}{r_2^2}|r_2 - r_1| + \frac{m_1+m_2+m_3}{r_3^2}|r_3 - r_2| + \ldots$$

4. iteratively remove unbound particles

1. obtain initial set of particles and determine $M_{\rm halo}$ and $R_{\rm halo}$

2. calculate $\varphi_0 = G\left(\frac{M_{vir}}{R_{vir}} + \int_0^{R_{vir}} \frac{M(<r')}{r'^2} dr'\right)$

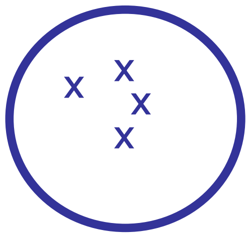
3. while looping over all particles check $v_i > v_{\rm esc}(r_i) = \sqrt{2|\varphi(r_i)|}$

4. using $\varphi(r_i) = G\int_0^{r_i} \frac{M(<r)}{r^2} dr - \varphi_0$

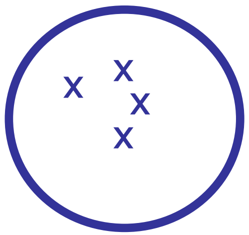
5. bound particles define a new set of initial particles for $M_{\rm halo}$ and $R_{\rm halo}$

$\Rightarrow$ start from 2. again and repeat until no further unbound particles...

4. iteratively remove unbound particles

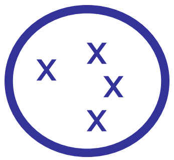


5. determine halo edge (final)



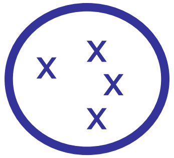
(re-adjust radius)

5. determine halo edge (final)



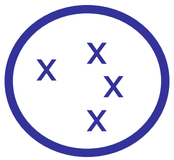
6. eventually determine halo properties

$$R_{\text{halo}} = \begin{cases} \text{the point where the density profile} \\ \text{of bound particles drops below } \Delta\rho_{ref} \\ \\ \text{distance to farthest bound particle within “tidal radius”} \end{cases}$$



6. eventually determine halo properties

$$R_{\rm halo} = \begin{cases} \text{the point where the density profile} \\ \text{of bound particles drops below } \Delta\rho_{ref} \\ \\ \text{distance to farthest bound particle within "tidal radius"} \end{cases}$$

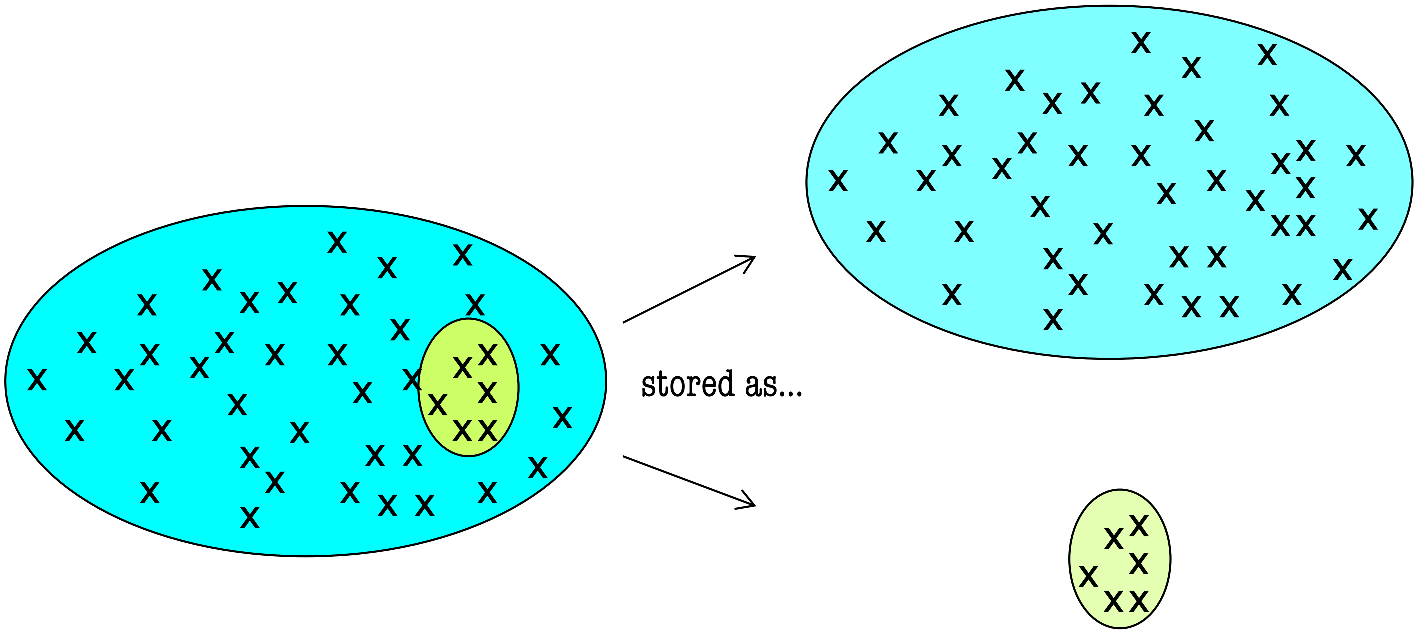


all other halo properties are based upon particles

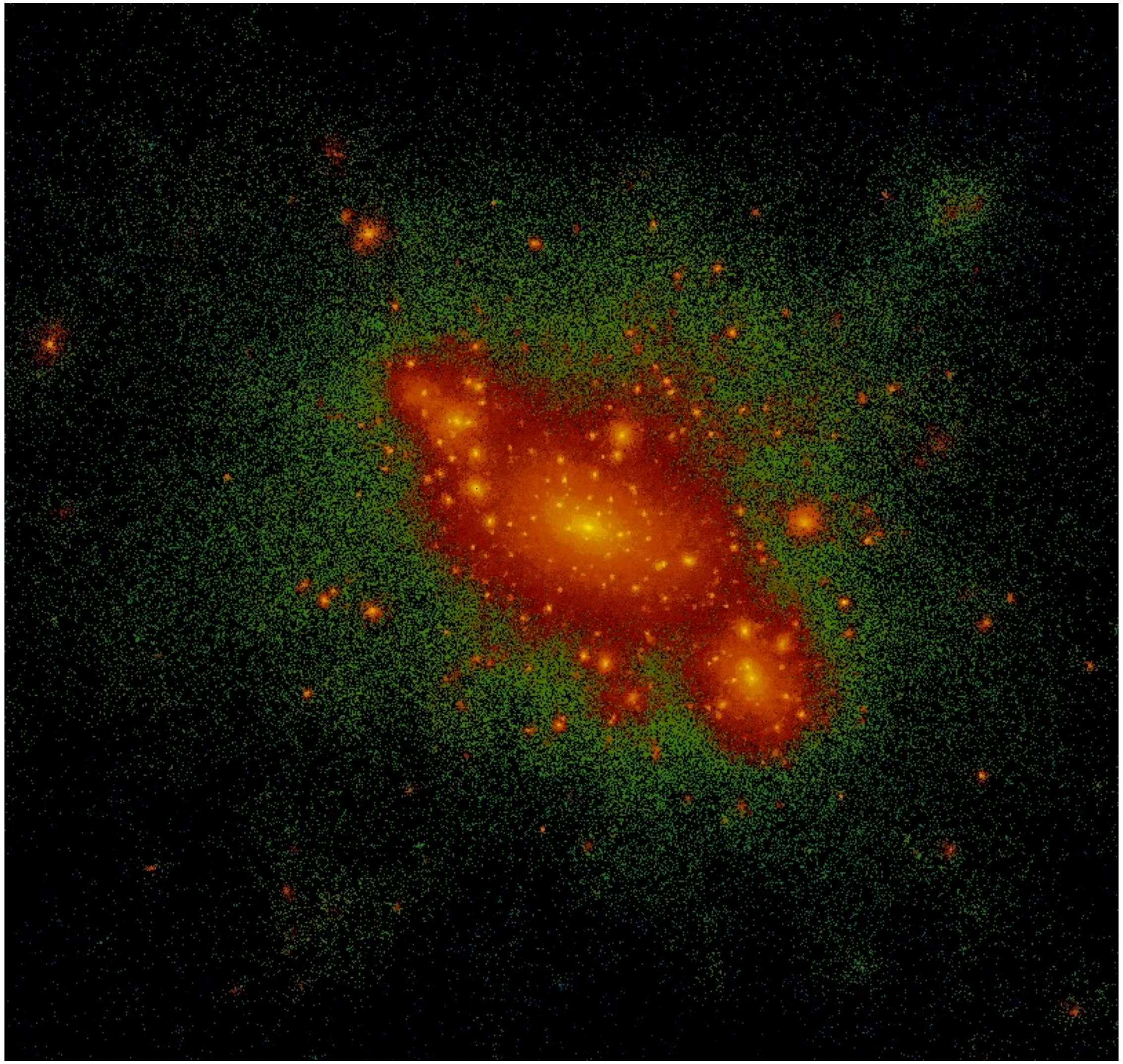
inside sphere of radius $R_{\rm halo}$!

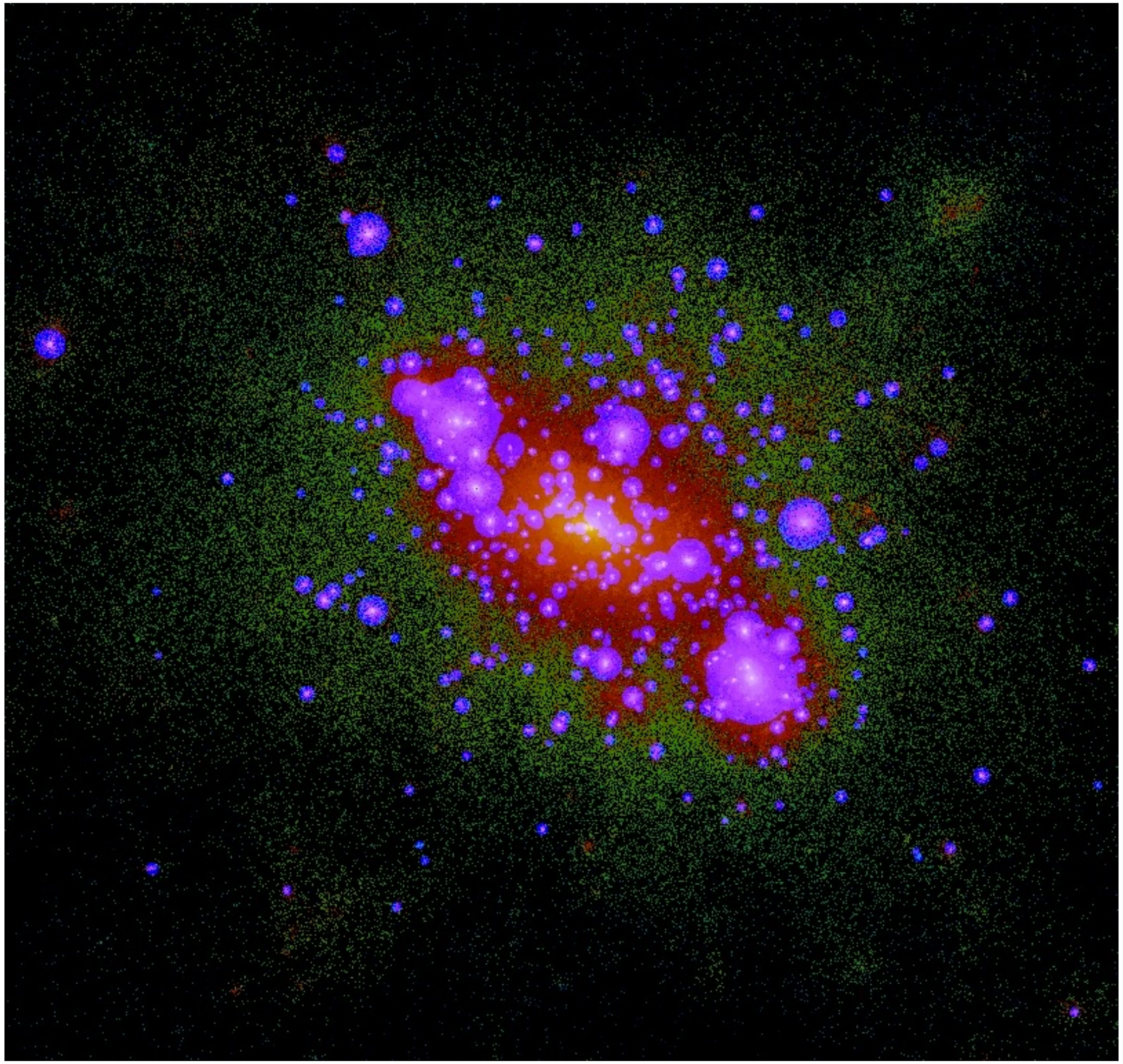
6. eventually determine halo properties

- please note that subhalo particles are included in the host halo, too!



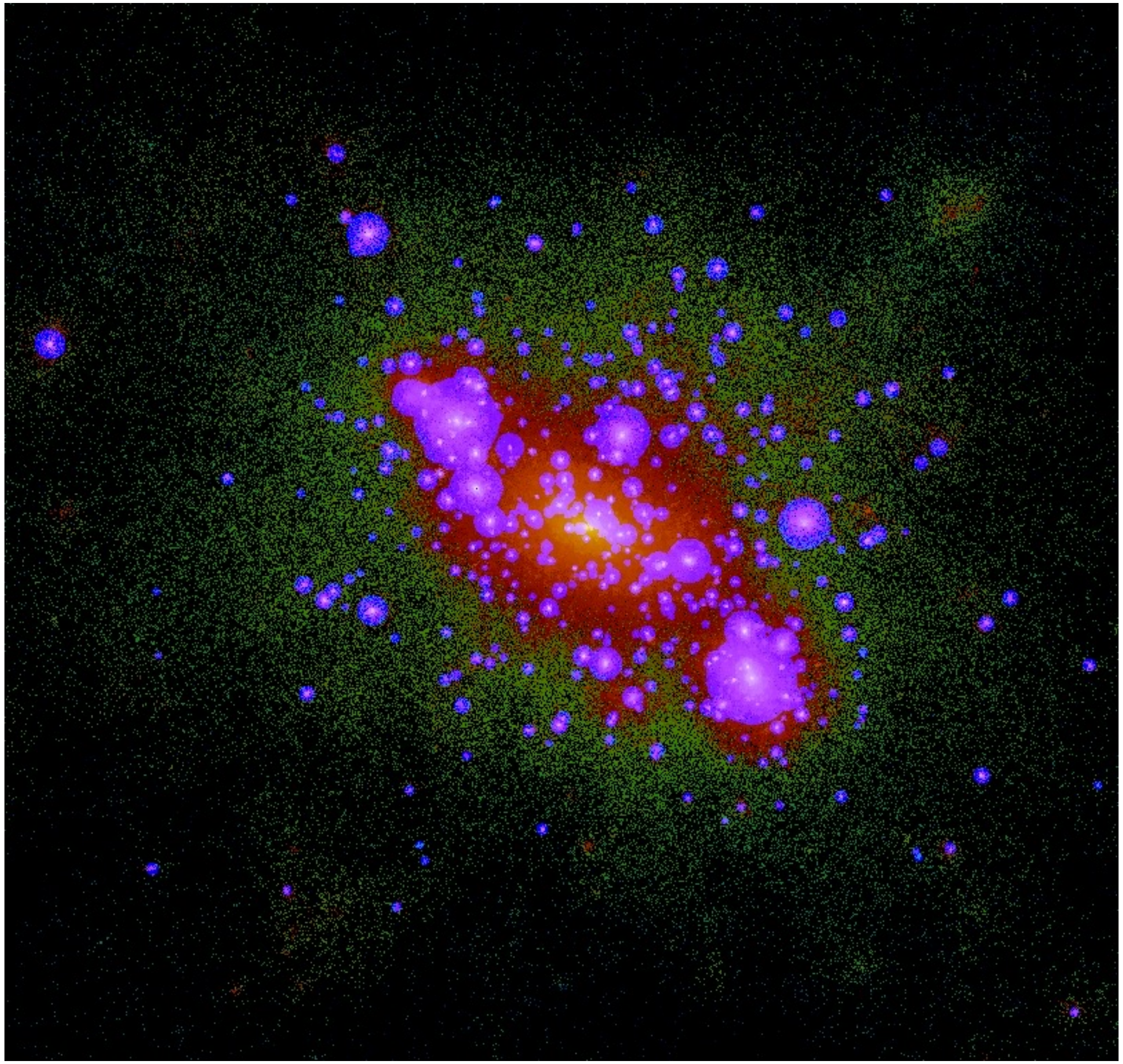
- subhalos are contributing to the integral properties of their hosts





"host" halo not shown for clarity

AHF naturally find haloes, sub-haloes, sub-subhaloes, ...



"host" halo not shown for clarity

How to Compile?

(DEFINEFLAGS)

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

```
           Makefile.config
           analysis/
           bin/
ahf/       convert/
           documentation/
           src/
           tools/
```

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

`ahf/`
- `Makefile.config`
- `analysis/`
- `bin/`
- `convert/`
- `documentation/`
- `src/`
- `tools/`

only ever edit this `Makefile.config`, none of the other Makefiles!

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

```
ahf/
    Makefile.config
    analysis/
    bin/
    convert/
    documentation/
    src/
    tools/
```

`analysis/` ← contains what is consider analysis tools, e.g. `MergerTree.c`, etc...

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

```
ahf/
    Makefile.config
    analysis/
    bin/            <- all binaries will be placed here
    convert/
    documentation/
    src/
    tools/
```

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

`ahf/`

- `Makefile.config`
- `analysis/`
- `bin/`
- `convert/`
- `documentation/`
- `src/`
- `tools/`

`convert/` contains all sorts of conversion tools, e.g.

- `ramses2gadget.c` (courtesy Timur Doumler)
- Hdf2GADGET/ (courtesy Ofer Metuki)
- tostd/ (courtesy Doug Potter)
- etc.

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

```
ahf/
    Makefile.config
    analysis/
    bin/
    convert/
    documentation/
    src/
    tools/
```

empty, please fill it by downloading the documention from the web... → `documentation/`

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

```
ahf/
    Makefile.config
    analysis/
    bin/
    convert/
    documentation/
    src/        <- the heart-and-soul of AHF
    tools/
```

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

```
ahf/    Makefile.config
        analysis/
        bin/
        convert/
        documentation/
        src/
        tools/    <-- tools to make life easier...
```

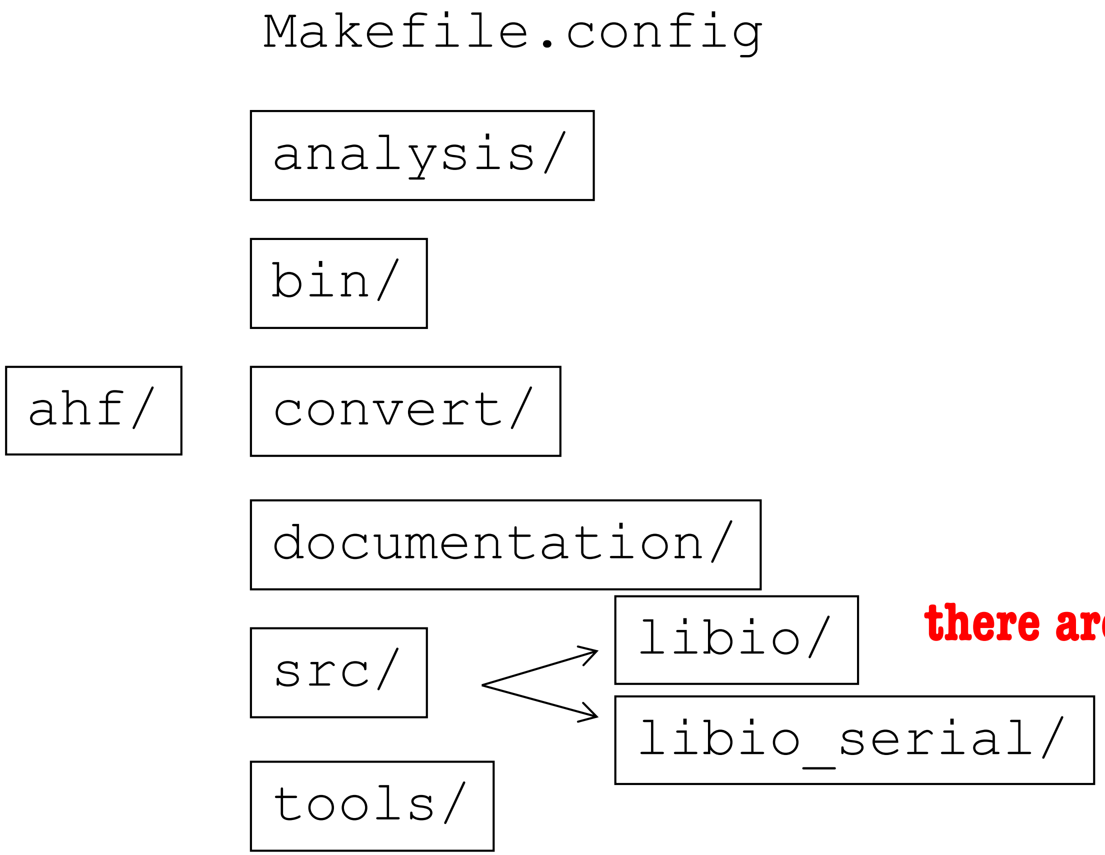
- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:

```
ahf/    Makefile.config
        analysis/
        bin/
        convert/
        documentation/
        src/
        tools/
```

if you plan to use *AHF* as a black-box the only files you ever need to touch are...

`ahf/Makefile.config`
`ahf/src/param.h`
`ahf/src/define.h`

- after unpacking the tarball `ahf-v1.0.tgz` you end up with the following directory layout:



there are two different libraries supporting I/O!

`libio/`

- this library is used for everything in src/, i.e. all of ***AHF***

AHF uses libio/ and nothing else!

`libio_serial/`

- this library is the one used with all simuXYZ tools in analysis/ and tools/
- the file type must be specified prior to compilation using DEFINEFLAGS
- *you may like to get in touch with us if want to use this ;-)*

`Makefile.config` and `src/define.h`

All features of ***AHF*** are controlled via *#ifdef FEATURE* in the source code and hence can be activated by either

-DFEATURE in the `Makefile.config`

or

#define FEATURE in `src/define.h`

`src/param.h`

Some parameters controlling the behaviour of ***AHF*** are to be set here...

How to Compile?

`Makefile.config`

- Makefile.config

Please note that you need to generate a Makefile.config and should **not** touch the actual Makefile found in the top level (or any other level of the source hierarchy!) directory at all!

All your favourite flags and definitions could go into your Makefile.config and we provide a sample to be used at your leisure...

OpenMP or MPI code?

Besides of the option to switch on/off various features via –DFEATURE you should also choose your system. There are three standard configurations that should work on most common machines:

- "Standard OpenMP"

  choose this for the OpenMP version

- "Standard MPI"

  choose this for the MPI version

- "Standard MPI+OpenMP"

  choose this for the MPI+OpenMP hybrid version

OpenMP and MPI work nicely together and are not mutually exclusive!

A simple `make` will then produce the respective code...

DEFINEFLAGS

How to Compile? — DEFINEFLAGS

MULTIMASS	simulations with particles of multiple masses
BYTESWAP	force byteswap when reading data
GAS_PARTICLES	simulation includes gas and/or star particles
WITH_MPI	MPI domain decomposition
WITH_OPENMP	OpenMP for-loops
REFINE_BARYONIC_MASS	weigh baryonic particles by m_b/m_{DM} for number density used to trigger refinemens
DARK_ENERGY	allow for non-standard $H(z)$ used for $\rho_{\rm crit}$=3H^2/8πG
AHFlean	reduce memory usage
AHFsubstructure	write AHF_substructure file
AHFdisks	write AHF_disks file
AHFsorthalosbymass	sort halos by mass (instead of # of particles) when writing output files
AHFaddDMonlyproperties	additionally calculate DM only properties in SPH simulations (not writing to file yet!)
AHFdmonlypeaks	only use DM particles for the determination of halo centres
AHFnoHubbleDrag	ignore $+Hr$ term for velocities
AHFundoPositionShiftAndScale	write haloes' position in the same frame as found in the simulation input file
AHFnoremunbound	do not perform unbinding
AHFignore_ugas	ignore thermal gas energy
AHFreducedinertiatensor	use reduced moment of intertia tensor for shapes
AHFshellshape	use particles in shell (rather than in sphere) to calculate inertia tensor
AHFvmbp	write velocity of most-bound particle to AHF_halos
AHFdmonly_Rmax_r2	base Rmax and r2 determination on dark matter profile only
AHFparticle_Rmax_r2	use all particles (instead of binned profile) to obtain Rmax and r2
AHFptfocus	only use a certain particle species for analysis
AHFrfocus	focus analysis on spherical region only
AHFsplinefit	use spline-fit to determine Vmax, etc.
AHFmaxdenscentre	use cell with maximum density as halo centre
AHFpotcentre	use potential weighted centre as halo centre
AHFgeomcentre	use geometrical centre as halo centre
AHFcomcentre	use centre-of-mass of particles on finest refinement as halo centre
AHFnewHaloIDs	halo IDs will be generated using number of particles in halo and its position
AHFmixHaloIDandSnapID	construct a haloid that combines the actual haloid and the snapshotid
AHFbinary	write AHF_halos, AHF_profiles, and AHF_particles in binary format
AHFcentrefile	write file containing all potential halo centres
AHFgridtreefile	write file containing the full grid-tree information
AHFcR1	additionally write halo concentrations as defined by Wang et al. (arxiv:2310.00200) into AHF_halos

HOW TO COMPILE? — DEFINEFLAGS

DPhalos	write file containing all potential halos prior to ahf_halos.c
TIPSY_ZOOMDATA	shifts TIPSY particles by half-a-boxsize when reading
TIPSY_PARTICLE_ORDERING	stick to ordering of particles in TIPSY file
CUBEP3M_WITH_PIDS	for CubeP3M files that contain particle IDs
PARDAU_DISTANCE	use sub-grid with closest distance to follow host branch
PARDAU_NODES	use sub-grid with most nodes to follow host branch
PARDAU_PARTS	use sub-grid with most particles to follow host branch
NCPUREADING_EQ_NFILES	speeds up reading of multiple GADGET files immensely, but requires what it says!

the following flags exist, but are not described here in detail; if you are interested in them, please do get in touch!

PERIODIC	toggle periodic boundary conditions
VERBOSE/2	verbose execution
FOPENCLOSE	do not open multiple GADGET files all at the same time, but one after the other (default!)
BCASTHEADER	only one MPI task will read the GADGET header and then broadcast information (defualt!)
CHECK_RLIMIT_NOFILE	check whether number of allowed file descriptors exceeds required number
SUSSING2013	writes *_particles files compliant with the format for the Sussing Merger Trees workshop (Note, this flag overwrites AHFnewHaloIDs! Also works perfectly fine with WITH_MPI!)

- general remarks

The DEFINEFLAGS (i.e. *#ifdef FEATURE* in the code) can either be activated by using

-DFEATURE in the Makefile.config

or putting the desired

#define FEATURE into src/define.h

Makefile.config:

You will note that the Makefile.config already comes with a set of DEFINEFLAGS predefined for various projects/snapshots; and I recommend to keep track of your features in a similar way (it makes life easier when coming back to re-analyse the simulation after a vacation or any other break...)

define.h:

Please check define.h **very** carefully as some features are mutually exclusive and are being switched on or off depending on some other features!

- classes of DEFINEFLAGS
  - general features
  - *AHF* features
  - misc. features

rember that either -DFEATURE in `Makefile.config` or #define FEATURE in `define.h` will switch it on; however, we refer to the feature from now on as "#define FEATURE"...

- general features

#define MULTIMASS

  - if your simulation contains particles with different masses you **must** switch this feature on! Otherwise ***AHF*** will not allocate the array to store the masses and fail gloriously...

- general features

#define BYTESWAP

  - forces a byteswap when reading the simulation binary file
  - you need to use this flag when…
    - » your data is little_endian but your analysis machine big_endian
    - » your data is big_endian but your analysis machine little_endian

Note: this feature is obsolete when analysing **GADGET** files with the -DNEWSTARTRUN version

- general features

#define GAS_PARTICLES

  - in case you are supplying also gas and star particles ***AHF*** will add additional columns to the *.AHF_halos and *.AHF_profiles files containing information about the properties of the gas and stellar content of each halo alone...

This feature should *definitely* be used whenever you are dealing with simulations including baryons (gas and/or stars)!

Note: you cannot switch off this feature for star particles, i.e. GAS_PARTICLES switches it on for both!

- general features

#define WITH_MPI

  - now ***AHF*** can be run on a distributed memory machine
  - please note that this requires additional parameters in the parameter input file `AHF.input` used when starting the code!

- general features

#define WITH_OPENMP

  - the processing of individual halos will be cast to different threads
  - define # of threads via OMP_NUM_THREADS environment variable
  - works perfectly together with WITH_MPI

- general features

#define REFINE_BARYONIC_MASS

- normally ***AHF*** uses the number density of particles to refine a cell
- this flag will switch to using the baryonic mass density to refine a cell, i.e. they only count as m_b/m_{DM} when calculating the number density
- Note, dark matter particles still contribute via number density

- ***AHF*** features

#define DARK_ENERGY

  - use tabulated values for $H(z)$ for the calculation of $\rho_{crit}=3H^2/8\pi G$
  - this feature is not fully public yet; if you like to use it, get in touch!

- ***AHF*** features

#define AHFlean

- this removes (hopefully) all memory associated with the simulation code ***AMIGA*** not related to ***AHF***
- we find that the memory consumption goes down significantly and hence made this feature STANDARD

- ***AHF*** features

#define AHFsubstructure

- writes an additional file `AHF_substructure` containing information about which halo is a subhalo of what host
- Note, the file `AHF_halos` will already contain a pointer to its host halo as well as the number of its subhaloes
- **please check *very carefully* the limitations of both this file as well as the information given in `AHF_halos` explained on page 173++**

- ***AHF* features**

#define AHFdisks

- writes an additional file `AHF_disks` containing information about potential (baryonic) disks in the centre of haloes

- **if you are interested in this feature, please see 'format of output files', but also do get in touch as it is still under development**

- ***AHF*** features

#define AHFsorthalosbymass

- the output in the halo catalogues is by default ordered by the number of particles in a halo
- use this feature to order the output by halo mass

- ***AHF*** features

#define AHFaddDMonlyproperties

- for simulations including gas and star particles some properties are calculated using only these particles (when using GAS_PARTICLES)

- this feature then also calculates properties based solely on dark matter particles

- NOTE: these properties are not yet written to file and you would need to modify src/libahf/ahf_io.c yourself to dump this information! (or kindly ask us...)

- ***AHF* features**

#define AHFdmonlypeaks

  - in some SPH simulations you might end up with baryon-only clumps that are rather unphysical for reasons of SPH implementation; to avoid them only use the DM to locate potential halo centres

- ***AHF*** features

#define AHFnoHubbleDrag

  - will not consider the Hubble drag +***H*r*** during unbinding

- ***AHF*** features

#define AHFundoPositionShiftAndScale

- write halo positions in original units as found in the input file
- Note, normally ***AHF*** shifts all positions to lie inside a cubic box [0,B]

- ***AHF*** features

#define AHFnoremunbound

- skips the unbinding procedure
- Note, this can also be achieved by setting `VescTune` in `AHF.input` to some ridiculously high value

- ***AHF* features**

#define AHFignore_ugas

- ignore the gas thermal energy when unbinding
- Note, standard practice for AHF is to use

$$e_{gas} = \phi + \frac{1}{2}v^2 + u$$

for each gas particle entering the unbinding procedure where

$e_{gas} =$ total specific energy

$\phi =$ gravitational potential

$v =$ gas particle's velocity

$u = \frac{3}{2}\frac{k_B}{m}T$ gas particle's thermal energy

- ***AHF*** features

#define AHFreducedinertiatensor

  - uses the reduced moment of inertia tensor to determine halo shapes

- ***AHF* features** #define AHFshellshape

  - base calculation of moment of inertia tensor on particles inside radial shells rather than all particles in sphere.

  - the overall shape of the halo is now the shape of the last profile bin

  - Note, there is a parameter AHF_MINPART_SHELL controlling the behaviour in `src/param.h` setting the minimum number of particles inside a shell for this calculation only.
    Further, a lot of the small mass haloes will now not have any shape measure anymore as there will be too few particles in the shells.

- ***AHF*** features

#define AHFvmbp

  - write the velocity of the most bound particle as additional columns into `AHF_halos`

- ***AHF*** features

#define AHFdmonly_Rmax_r2

  - calculate Rmax and r2 based upon dark matter only
  - Note, Vmax will still use all matter incl. gas and stars!

- ***AHF* features**

#define AHFparticle_Rmax_r2

- instead of using a binned profile to find Rmax and r2 this feature will use the full list of particles in each halo, i.e. no binning
- Note, to avoid particles right in the centre leading to M/r=inf for the circular velocity curve there is a new parameter in src/param.h: AHF_Rmax_r2_NIGNORE which will ignore the innermost particles

- ***AHF* features** #define AHFptfocus=*value*

• only keeps particles of a certain kind for ***AHF*** analysis

• set the "particle-type-to-keep" as follows:

  0 = gas particles

  1 = dark matter particles

  4 = star particles

• if you have more than one dark matter type, please consult `main.c` where this feature is to be found and/or get in touch with us...

- ***AHF*** features

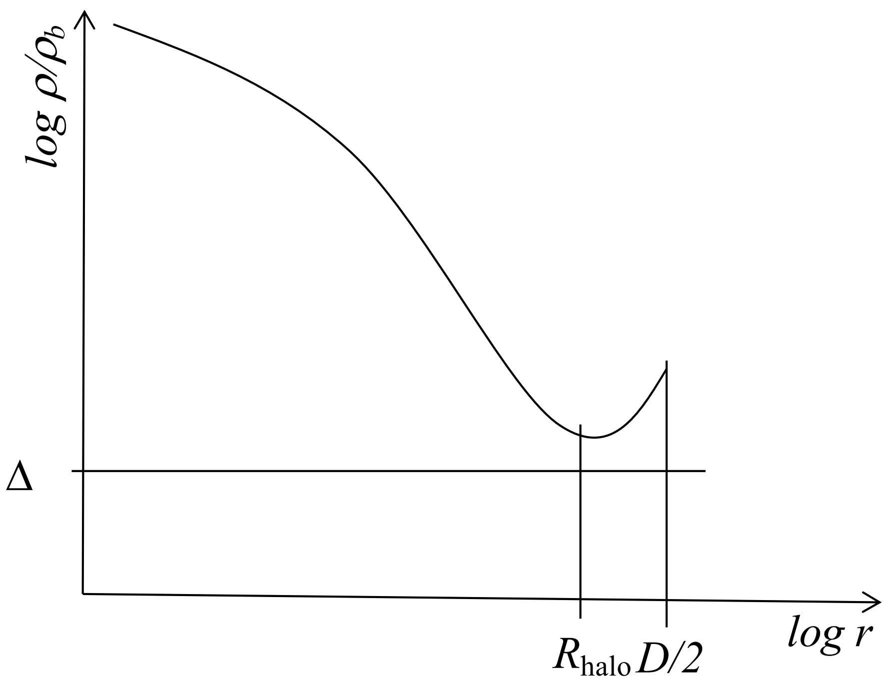
#define AHFrfocus

  - remove all particles outside a sphere whose particulars are defined in src/param.h

- ***AHF*** features

#define AHFsplinefit

- uses a splinefit routine to determine R_{halo}
- use with care as this may not work for low-mass haloes with too few bins!



- ***AHF*** features

#define AHFmaxdenscentre

- per default ***AHF*** determines the prospective halo centre as the density-weighted centre of the "end-leave" in the AMR grid tree

- this feature rather uses that cell in the end-leave grid with the highest density value as prospective halo centre

- ***AHF*** features

#define AHFpotcentre

- per default ***AHF*** determines the prospective halo centre as the density-weighted centre of the "end-leave" in the AMR grid tree
- this feature rather uses that cell with the lowest value of the potential as the potential halo centre
- **Note:** this feature requires substantially more time for ***AHF*** to run as it solves for the potential on the complete AMR hierarchy!

- ***AHF*** features

#define AHFgeomcentre

  - per default ***AHF*** determines the prospective halo centre as the density-weighted centre of the "end-leave" in the AMR grid tree

  - this feature rather uses the geometrical centre of the refinement patch

- ***AHF*** features

#define AHFdensrecovery

  - this ensures that the density on all refinement levels is 100% correct
  - you only require this when using AHFpotcentre or AHFmaxdenscentre

- ***AHF*** features

#define AHFcomcentre

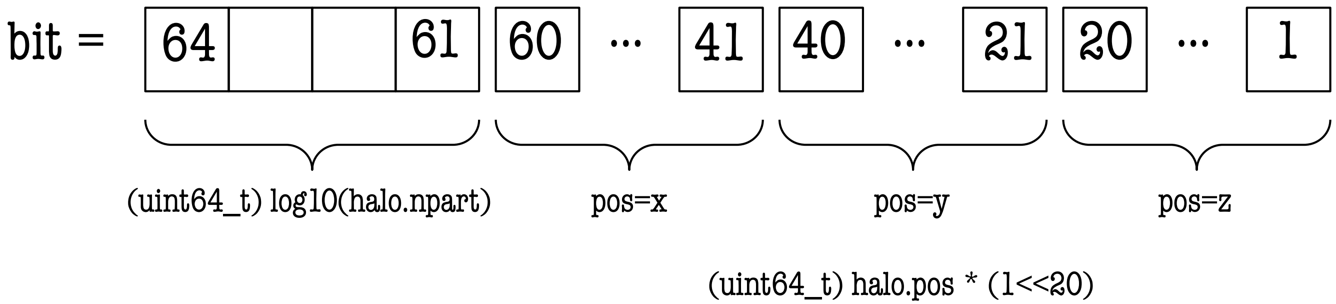
  - per default ***AHF*** determines the prospective halo centre as the density-weighted centre of the "end-leave" in the AMR grid tree
  - this feature rather uses the centre-of-mass of the particles encompassed by the refinement patch

some trial-and-error with these AHF***centre **flags indicated that** AHFcomcentre **gives the best results for subhaloes…at least for our simulations…**

- ***AHF* features** #define AHFnewHaloIDs

  - instead of using consecutive numbering for the halo IDs this feature will assign to each halo a (unique) ID based upon its number of particles and its position.

  - we are using a 64 unsigned integer to store the ID with the bits assigned as follows:



  - this feature is switched on by default for the MPI version!

- ***AHF*** features

#define AHFmixHaloIDandSnapID

- instead of using consecutive numbering for the halo IDs this feature will assign to each halo a (unique) ID based upon its number and the snapshot it is located in.

- this requires one additional parameter to the AHF.input file, i.e. the snapID.

- ***AHF*** features

#define AHFbinary

- writes the files

  `AHF_halos`
  `AHF_profiles`
  `AHF_particles`

  in binary format

- Note, the files `AHF_substructure` and `AHF_particlesSTARDUST` will still be written in ASCII.

- check the IDL routines ReadAHFbinaryfile and ReadAHFbinaryprofile in `analysis/IDL/util.pro` how to read the binary files...

- ***AHF*** features

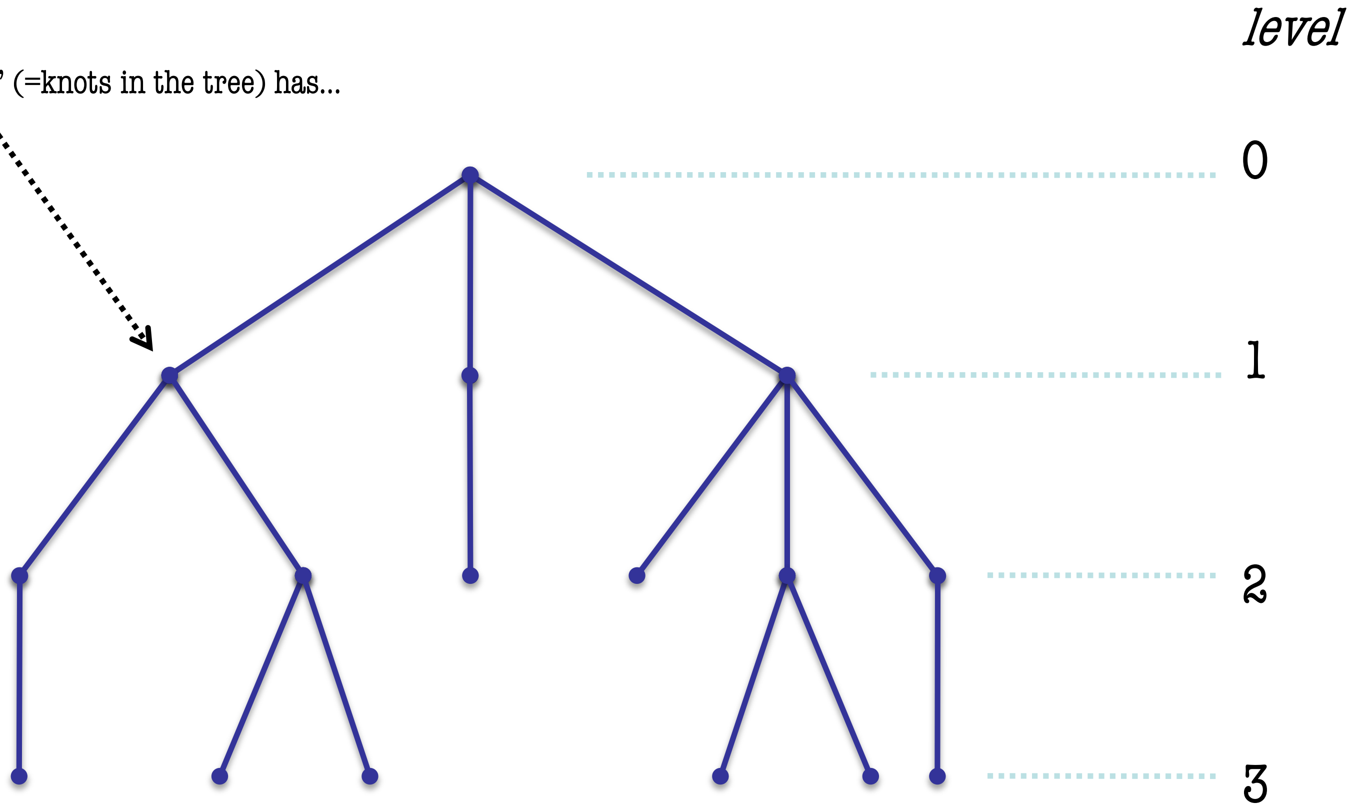
#define AHFcentrefile

  - writes an additional file containing all the prospective halo centres, i.e. the density peaks found in the simulation

- ***AHF*** features

#define AHFgridtreefile

each of these "patches" (=knots in the tree) has...
- centre x,y,z
- number of particles
- number of cells
- number of daughters



compile with –DAHFgridtreefile and you the codes stops after writing a file with the following information:

level	x	y	z	npart	ncells	ndaughter

for each patch on all levels!

- ***AHF*** features

#define AHFcR1

• we follow the ideas of Wang et al. (arxiv:2310.00200) in calculating the halo concentration.

• with this switched on there will be two additional columns in AHF_halos containing both c (as found by inverting Eq.(4) below) and the R1 value itself.

Here $\rho(r)$ is the radial density profile and $r_{\rm vir}$ is the halo radius, which is usually defined as the radius within which the enclosed mean density just exceeds some chosen value. The dimensionless first moment of the density distribution, R_1, can be defined as

$$R_1 = \frac{1}{M_{\rm vir} r_{\rm vir}} \int_0^{r_{\rm vir}} 4\pi r^3 \rho(r) dr \, , \tag{3}$$

which can be expressed analytically for an NFW profile as

$$R_1 = \frac{c - 2\ln(1+c) + c/(1+c)}{c\,[\ln(1+c) - c/(1+c)]} \, . \tag{4}$$

- ***AHF*** features

#define DPhalos

  - writes an additional file containing all the unprocessed list of halos
  - Note, this likely only serves those who do some debugging of ahf_gridinfo.c and is of no real use for the end-user ... yet.

- ***AHF*** features

  - there are three features that control halo vs. subhalo treatment:

```
#define PARDAU_DISTANCE
#define PARDAU_NODES
#define PARDAU_PARTS
```

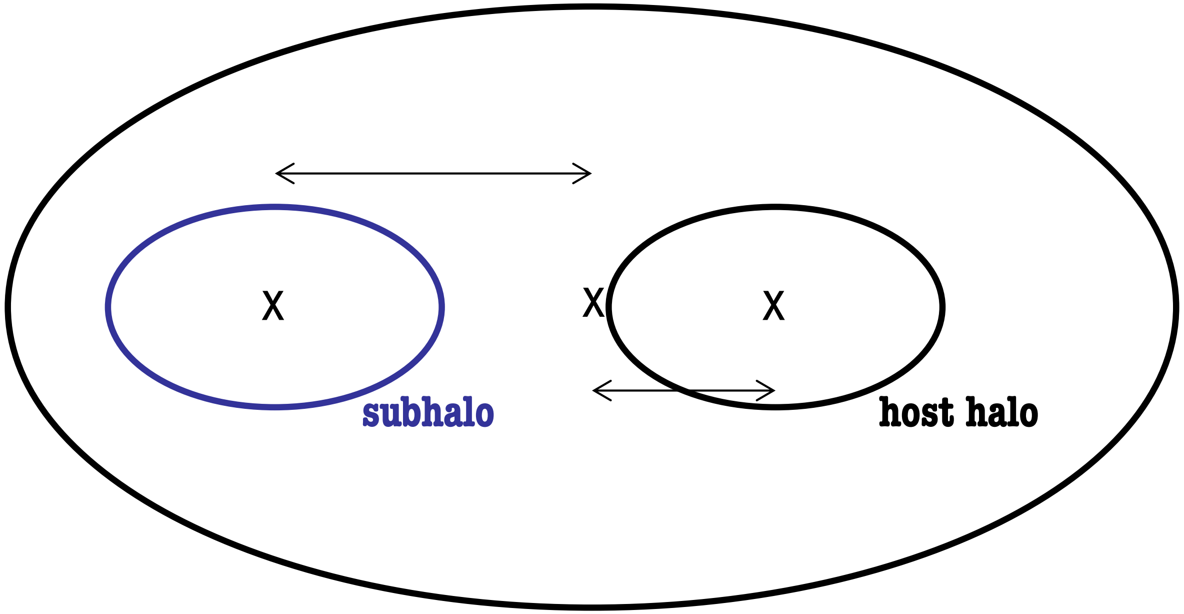
  - they control the classification into halo, subhalo, sub-subhalo, etc.

  - a major merger of two nearly equal mass objects can cause a lot of trouble and hence experimenting with this feature in that case may help?!

- *AHF* features

#define PARDAU_DISTANCE

- parent-daughter assignment is done by distance

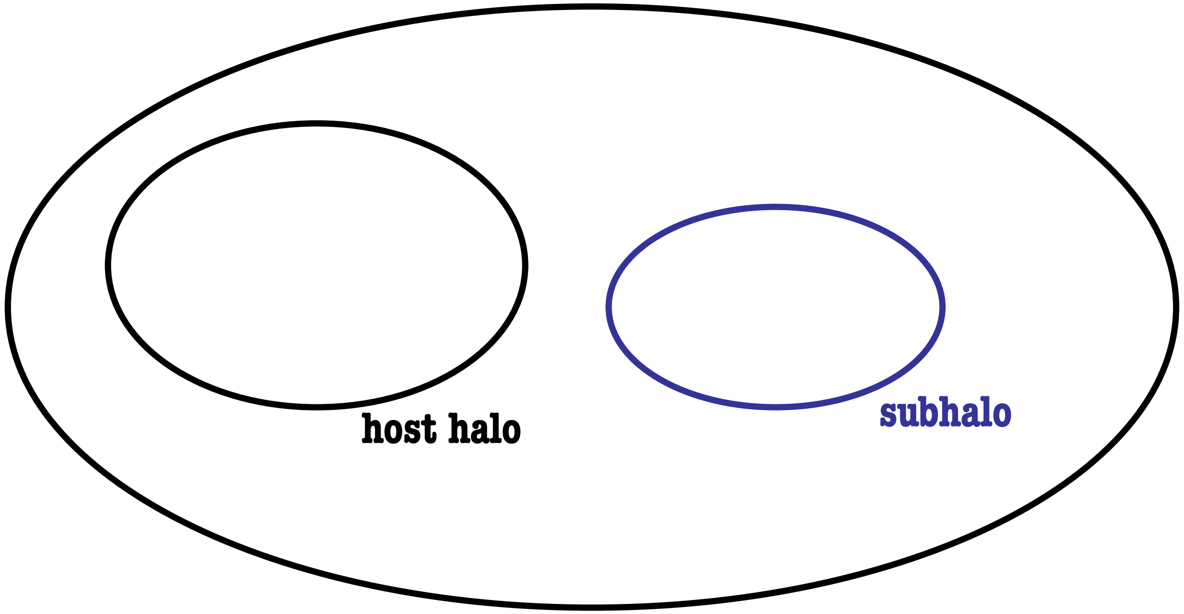


- those parent-daughter grids with the smallest distance are being tagged as "trunk" in the AMR grid tree

- ***AHF*** features

#define PARDAU_NODES

- parent-daughter assignment is done by number of cells

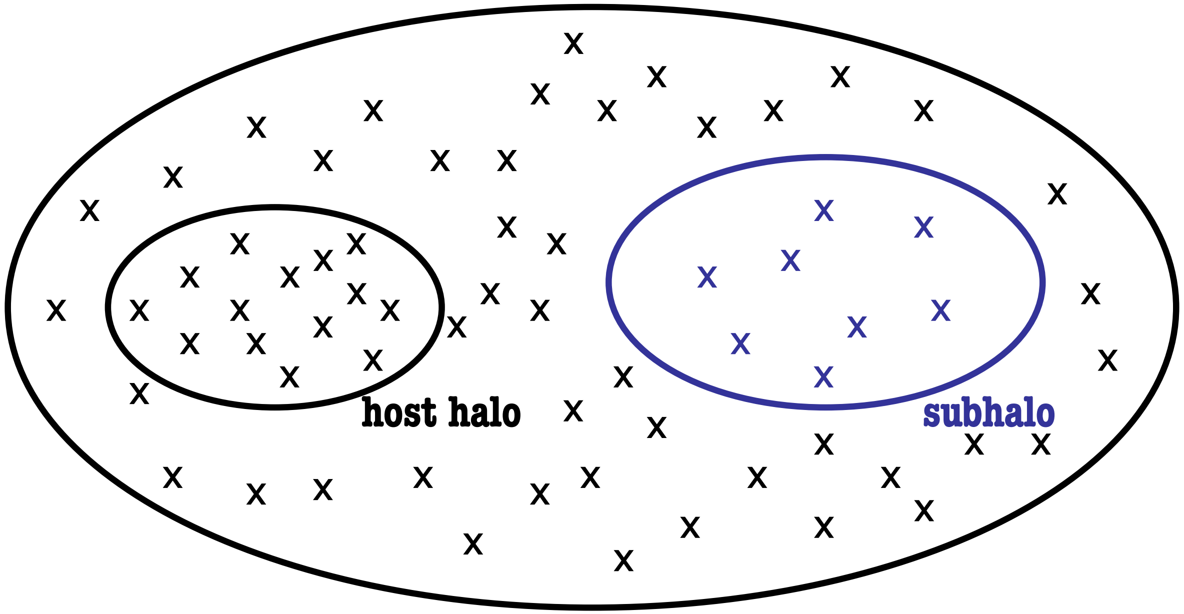


- the largest daughter grid is being tagged as "trunk" in the AMR grid tree

- ***AHF*** features

#define PARDAU_PARTS

- parent-daughter assignment is done by number of particles



- the daughter grid with the most particles is being tagged as "trunk" in the AMR grid tree
- this daughter grid is the most likely candidate for further refinement and encompassing the highest density peak, respectively

switched on by default (cf. `define.h`)

AHF - AMIGA's HALO FINDER

- misc. features

#define TIPSY_ZOOMDATA

  - shifts TIPSY particles by half-a-boxsize when reading

- misc. features #define TIPSY_PARTICLE_ORDERING
  - to assign unique IDs across multiple simulation snapshots (at least for the dark matter particles) ***AHF*** moves the DM particles to the beginning of the particle memory block giving them IDs from 0 to N_{dm}. Using this feature the particle IDs are now set according to the position in the TIPSY file instead keeping the order stars, dark matter, gas!

- misc. features

  #define CUBEP3M_WITH_PIDS

  - reads in particle IDs from CubeP3M files

- misc. features

#define NCPUREADING_EQ_NFILES

  - this will speed up reading of multiple GADGET beyond belief!
  - but you ***must*** use as many MPI tasks to read as there are files!
  - the analysis can be done using a different number of tasks though...

How to Compile?

src/param.h

- overview

AHF_MINPART_GAS	minimum number of gas particles used when calculating gas shapes, etc.
AHF_MINPART_STAR	minimum number of star particles used when calculating gas shapes, etc.
AHF_MINPART_SHELL	minimum number of particles in shell, only used with AHFshellshape
AHF_NBIN_MULTIPLIER	factor to increase the number of bins used for AHF_profiles
AHF_HOSTSUBOVERLAP	first level to be considered as credible to spawn subbaloes
AHF_rfocusX,Y,Z,R	spherical region used with -DAHFrfocus (cf. define.h), units in Mpc/h
AHF_MIN_REF_OFFSET	offset for first refinement to be used
AHF_HIRES_DM_WEIGHT	weight of high-resolution dark matter particle in internal units
MIN_NNODES	smallest allowed grid block

- ***AHF*** behaviour

AHF_MINPART_GAS

  - for simulations containing gas particles ***AHF*** also calculates certain properties based upon the gas particles alone; this parameter sets the particle number limit for this sort of calculation (e.g. it does not make sense to calculate the moment of inertia tensor of 2 gas particles...)

- ***AHF*** behaviour

AHF_MINPART_STAR

  - for simulations containing star particles ***AHF*** also calculates certain properties based upon the star particles alone; this parameter sets the particle number limit for this sort of calculation (e.g. it does not make sense to calculate the moment of inertia tensor of 2 star particles...)

- ***AHF*** behaviour AHF_MINPART_SHELL

  - when using the feature AHFshellshape the code now checks whether there are more than AHF_MINPART_SHELL particles in a shell before it will actually diagonalize the moment of inertia tensor; if there are less particles, the eigenvalues and eigenvectors are set to zero.

- *AHF* behaviour

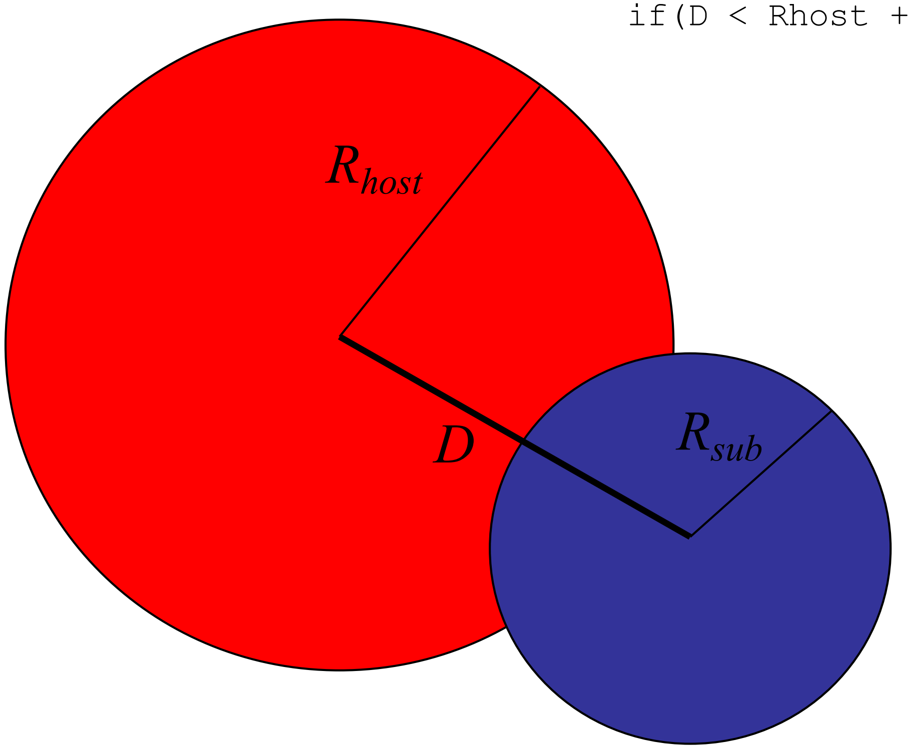
AHF_NBIN_MULTIPLIER

- in case you like to have more bins in the *.AHF_profiles file set this parameter to something larger than 1; you will end up AHF_NBIN_MULTIPLIER times more bins then...

- ***AHF*** behaviour — AHF_HOSTSUBOVERLAP

  - used with feature AHFsubstructure defining how far a halo is required to have entered the host to be considered a subhalo
  - the condition checked reads

```
if(D < Rhost + AHF_HOSTSUBOVERLAP*Rsub) subhalo=TRUE
```

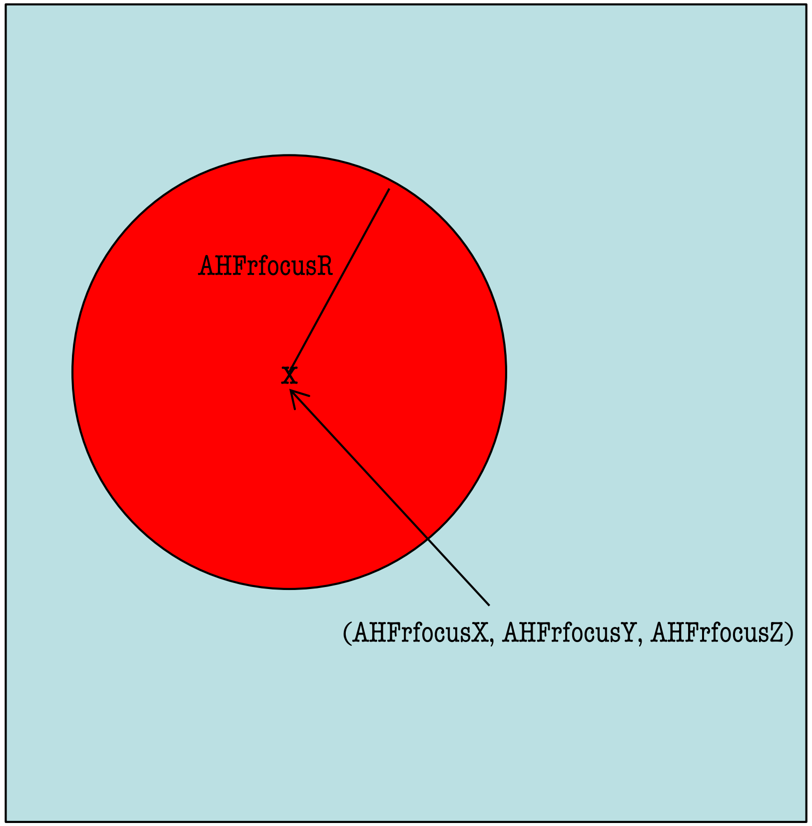


- **please check *very carefully* the limitations explained on page 173++**

▪ *AHF* behaviour

AHF_rfocusX,Y,Z,R

- when you switch on –DAHFrfocus these four parameters define a sphere whose particles are used for the analysis; all simulation particles outside will be removed from memory!
- the coordinates/radius are to be given in Mpc/h



full simulation box

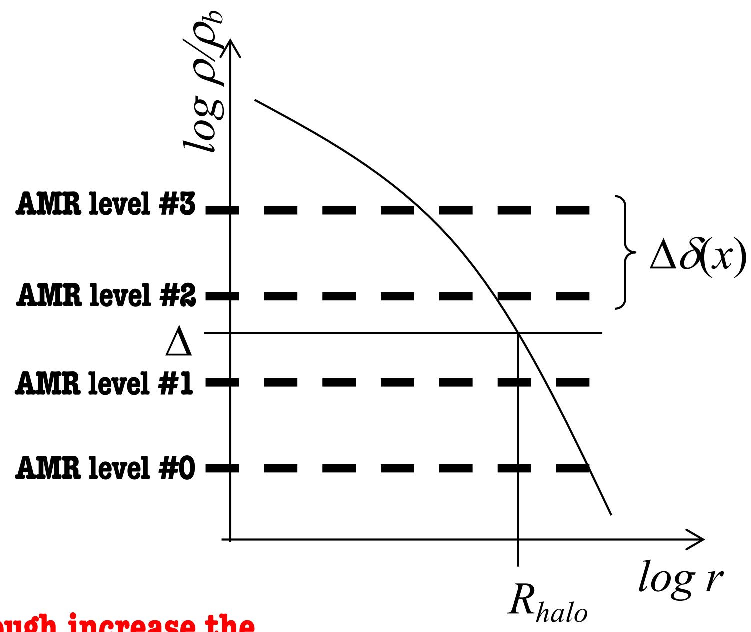
- ***AHF*** **behaviour** AHF_MIN_REF_OFFSET

  - ***AHF*** automatically determines the finest grid defining the isodensity contour closest to the virial overdensity criterion –> in the depicted example that would be AMR level #1
  - remember:

    $\Delta\delta(x)$: spacing of AMR isodensity contours as determined by refinement criterion
    Δ_{vir}: virial overdensity threshold as given by cosmology and redshift

⇓

...for the depicted example
AHF would only consider AMR levels
#(1+AHF_MIN_REF_OFFSET)
(and above) in the construction of halos!



While this flag may lead to host haloes that are too small it may though increase the performance dramatically when you are only interested in subhaloes! Decide for yourself... ;-)

- ***AHF*** behaviour

AHF_HIRES_DM_WEIGHT

  - facilitates the use and interpretation of fMhires: the value set here will be considered as the mass (*in internal units!!!!*) of the high-resolution dark matter particles

- ***AHF*** behaviour

MIN_NNODES

- sets the minimum number of cells per refinement grid, e.g. grids containing fewer cells are not considered trustworthy...
- controls refinements already on the `refine_grid.c` level

- general behaviour
  - all other parameters are even more technical and should not be tinkered with unless you know what you are doing...

How to Run?

- to execute ***AHF*** you need to generate an input file `AHF.input` explained right here right now...

- then simply type

```
$ AHF AHF.input
```

sit back, relax, and enjoy the show...

```
[AHF]
ic_filename          = /Where/Is/Your/Simulation/Snapshot
ic_filetype          = 61
outfile_prefix       = MyGreatSimulationAnalysedByAHF
LgridDomain          = 128
LgridMax             = 16777216
NperDomCell          = 5.0
NperRefCell          = 5.0
VescTune             = 1.5
NminPerHalo          = 20
RhoVir               = 0
Dvir                 = 200
MaxGatherRad         = 3.0
LevelDomainDecomp    = 6
NcpuReading          = 1

[GADGET]
GADGET_LUNIT         = 1.
GADGET_MUNIT         = 1e10
```

sample AHF.input

- [AHF]
  - mandatory block!

- [GADGET]
- [GIZMO]
- [TIPSY]
- [CUBEP3M]
- [ART]

optional blocks only required when analysing data of that specific file format

- [AHF] ic_filename

  - the name of the simulation input file incl. full path
  - Note, for multiple GADGET snapshots you need to include the trailing '.', i.e. 'snap_047.'!

- [AHF] ic_filetype
  - an integer number encoding the type of file you are analysing

0	***AMIGA***	
5	***ARES***	(***AMIGA*** – resimulation mode)
10	ASCII	
20	**CubeP3M**	(single snapshot)
21	**CubeP3M**	(multiple snapshots)
50	**GIZMO**	(single snapshot)
51	**GIZMO**	(multiple snapshots)
60	**GADGET**	(single snapshot)
61	**GADGET**	(multiple snapshots)
70	ART	(single snapshots)
71	ART	(multiple snapshots)
80	DEVA	(derived format)
81	DEVA	(native format)
90	TIPSY binary	

- [AHF] outfile_prefix

  - the output file names will be constructed as follows:

```
outfile_prefix.z?.???.AHF_halos
outfile_prefix.z?.???.AHF_profiles
outfile_prefix.z?.???.AHF_particles
outfile_prefix.z?.???.AHF_substructure
```

etc.

where z?.??? will be the redshift

- [AHF] LgridDomain
  - size of the domain grid in 1D (please use a power of 2)

- [AHF] LgridMax

  - size of finest refinement level to be generated (again, power of 2)
  - this parameter allows you to control the spatial resolution, i.e.

    $$\text{SpatialResolution} \simeq \text{BoxSize}/\text{LgridMax}$$

  - if you do not care about such a limitation set this to e.g. 2^30

- [AHF] NperDomCell
  - number of particles triggering a refinement on LgridDom
  - Note, this number can be a real number

- [AHF] NperRefCell
  - number of particles triggering a refinement on refinement grids
  - Note, this number can be a real number

- [AHF] VescTune
  - during the unbinding particles with speed in excess of

$$v > VescTune \ v_{esc}$$

are considered unbound

- [AHF] NminPerHalo
  - only halos containing at least NminPerHalo particles are written to file

  (**Note**: ***AHF*** internally stores and deals with all halos containing down to 2 particles...)

- [AHF] RhoVir

  - the halo edge is defined via the equation

$$\frac{M(<R_{halo})}{\frac{4\pi}{3}R_{halo}^3} = \Delta\,\rho_{ref}$$

  - this (integer) parameter defines what to use as ρ_{ref}:

    - RhoVir = 0 use $\rho_{vir} = \rho_{crit}(z)$
    - RhoVir = 1 use $\rho_{vir} = \rho_{back}(z)$

- [AHF] Dvir

  - the halo edge is defined via the equation

$$\frac{M(<R_{halo})}{\frac{4\pi}{3} R_{halo}^3} = \Delta \, \rho_{ref}$$

  - this parameter defines what to use as Δ:

    - Dvir < 0 let ***AHF*** calculate it using spherical top-hat-collapse
    - Dvir > 0 value to be used

- [AHF] MaxGatherRad

  - collecting particles about potential halo centres extends out to the "half-distance" of the closest refinement on the same level; this though limits this distance (in physical Mpc/h units!)

  - there is further an internal(!) switch that limits the distance to 1/4 of the boxsize in case you are analysing very small cosmological volumes...

  - Note, you will **not** find objects larger than MaxGatherRad!

- [AHF] LevelDomainDecomp

  - *MPI-only parameter*

  - it sets the grid that is used to do the domain decomposition:

$$L = 2^{\text{LevelDomainDecomp}}$$

AHF farms out the particles to the desired number of CPU's and then runs a serial version of the halo finder on each of these CPU's!

Therefore, it is important to create a boundary zone on each CPU that contains (replicates of the) particles from the neighbouring cells. In order **not** to cut a halo into pieces this boundary should at least be of order the virial radius of the most massive object expected to be found within the simulation.

LevelDomainDecomp hence needs to be carefully chosen,
i.e. $B/2^{\text{LevelDomainDecomp}}$ should be of order that virial radius!
(where B=box size of your simulation...)

- [AHF] NcpuReading
  - *MPI-only parameter*
  - number of CPU's reading your data
  - Note, this number can be different from the number of CPU's used to analyse the data!

- [GADGET]
  - GADGET_LUNIT = conversion factor of positions in file to Mpc/h
  - GADGET_MUNIT = conversion factor of masses in file to Msun/h

- [GIZMO]

  - GIZMO_LUNIT = conversion factor of positions in file to Mpc/h
  - GIZMO_MUNIT = conversion factor of masses in file to Msun/h

Note, GIZMO I/O requires HDF5 libraries to be installed
and for that reason you need to

a) adjust Makefile.config adding paths to your HDF5 installation
b) compile the code with –DWITH_HDF5

Further, for multiple GIZMO snapshots, AHF expects the suffix ".hdf5".
If you use a different one, please change this in the file

```
src/libio/io_mgizmo.c
```

searching for the line

```
f->numfiles = io_util_findfiles(f->path, f->stem, "%i", ".hdf5", &fnames);
```

- [TIPSY]

  - TIPSY_BOXSIZE = box size in Mpc/h
  - TIPSY_MUNIT = conversion factor of masses in file to Msun/h
  - TIPSY_VUNIT = conversion factor of velocities in file to km/sec
  - TIPSY_EUNIT = conversion factor of energies in file to $(km/sec)^2$
  - TIPSY_OMEGA0 = Omega0
  - TIPSY_LAMBDA0= Omega_Lambda0

Note, ***AHF*** will convert this information found in `AHF.input` to a file `tipsy.info` used by that part of the code dealing with the units ... this is not sophisticated, but works :-)

- [CUBEP3M]

  - CUBEP3M_BOXSIZE = box size in Mpc/h
  - CUBEP3M_NGRID = ask the experts...
  - CUBEP3M_NODES_DIM = ask the experts...
  - CUBEP3M_OMEGA0 = Omega0
  - CUBEP3M_LAMBDA0 = Omega_Lambda0

Note, ***AHF*** will convert this information found in `AHF.input` to a file `cubep3m.info` used by that part of the code dealing with the units ... this is not sophisticated, but works :-)

- [ART]

  - ART_BOXSIZE = box size in Mpc/h
  - ART_MUNIT = conversion factor of masses in file to Msun/h

Note, ***AHF*** will convert this information found in `AHF.input` to a file `art.info` used by that part of the code dealing with the units ... this is not sophisticated, but works :-)

(Format of) The Output Files

FORMAT OF OUTPUT FILES *.log

- the logfile contains all sorts of runtime information

- the most important for the end-user are
  - the first lines summarizing the supplied input values
  - the very last lines summarizing information about timing and memory consumption

- the parameter file lists the values of all parameters
- the most important for the end-user are the "simulation related values" summarizing the cosmology
- the file also lists all DEFINEFLAGS used during compilation
- Note, you can trigger writing of such a file using an ***AHF*** binary alone by typing

```
> AHF-v1.0 --parameterfile
```

- integral properties

(1) ID	halo ID	
(2) hostHalo[1]	ID of host halo, 0 (or -1) if halo itself is not a subhalo	
(3) numSubStruct[1]	number subhalos inside halo	
(4) Mhalo	mass of halo	[$M_\odot$/h]
(5) npart	number of particles in halo	
(6) Xc		
(7) Yc	position of halo	[kpc/h]
(8) Zc		
(9) VXc		
(10) Vyc	peculiar velocity of halo	[km/sec]
(11) VZc		

[1] **please carefully read pages 173++ to better understand these numbers and their limitations**

- integral properties

(12) Rhalo	halo radius	[kpc/h]
(13) Rmax	position of rotation curve maximum	[kpc/h]
(14) r2	position where ρr^2 peaks	[kpc/h]
(15) mbp_offset	offset between most bound particle and halo centre	[kpc/h]
(16) com_offset	offset between centre-of-mass and halo centre	[kpc/h]
(17) Vmax	maximum of rotation curve	[km/sec]
(18) v_esc	escape velocity at Rhalo	[km/sec]
(19) sigV	3D velocity dispersion	[km/sec]

FORMAT OF OUTPUT FILES *.AHF_halos

- integral properties

(20) lambda	spin parameter (Bullock et al. 2001 definition)	
(21) lambdaE	classical spin parameter (Peebles' definition)	
(22) Lx		
(23) Ly	(orientation of) angular momentum vector	|L|=1
(24) Lz		
(25) b	second largest axis of moment of inertia tensor	b/a
(26) c	third largest axis of moment of inertia tensor	c/a
(27) Eax		
(28) Eay	largest axis of moment of inertia tensor	|Ea|=1
(29) Eaz		
(30) Ebx		
(31) Eby	second largest axis of moment of inertia tensor	|Eb|=1
(32) Ebz		
(33) Ecx		
(34) Ecy	third largest axis of moment of inertia tensor	|Ec|=1
(35) Ecz		

- integral properties

(36) ovdens	overdensity at virial radius	
(37) nbins	number of bins used for the *.AHF_profiles file	
(38) fMhires	mass fraction in high resolution particles for zoom simus	
(39) Ekin	kinetic energy	$[M_\odot/h\ (km/sec)^2]$
(40) Epot	potential energy	$[M_\odot/h\ (km/sec)^2]$
(41) SurfP	surface pressure (Shaw et al. 2006 definition)	$[M_\odot/h\ (km/sec)^2]$
(42) Phi0	φ_0 (cf. unbinding procedure)	$[(km/sec)^2]$
(43) cNFW	NFW concentration (Prada et al. 2012 definition)	

IMPORTANT NOTES:

- all these values have been derived using **all** particles inside the halo, i.e. dark matter, gas, and star particles (if present)...
- if you switched on GAS_PARTICLES there will be additional columns listing a subset of these properties based upon gas and star particles alone
- all positions are in **comoving** coordinates
- the halo's bulk velocity is the peculiar velocity = a\dot{x}
- Vmax and v_esc are based upon GM/r and Phi, respectively, and have been obtained using physical distances

- radial profile of selected properties

(1) r	right edge of radial bin	[kpc/h]
(2) npart	number of particles inside sphere of radius r	
(3) M_in_r	mass inside sphere of radius r	[$M_\odot$/h]
(4) ovdens	$M(<r)/(4\pi r^3/3)$ / ρ_b	
(5) dens	$M(r)/(4\pi r^2 dr)$ / ρ_b with $M(r)$ = mass in current *shell*	
(6) vcirc	rotation curve	[km/sec]
(7) v_esc	escape velocity from material inside r-sphere	[km/sec]
(8) sigv	velocity dispersion of material inside r-sphere	[km/sec]
(9) Lx		
(10) Ly	angular momentum of material inside r-sphere	
(11) Lz		[$M_\odot$/h Mpc/h km/sec]

Note, a negative value for "(1) r" indicates that the results at that radius have not converged and are dominated by two-body collisions according to the criterion of Power et al. (2003)

- radial profile of selected properties

(12) b	second largest axis of moment of inertia tensor	b/a
(13) c	third largest axis of moment of inertia tensor	c/a
(14) Eax		
(15) Eay	largest axis of moment of inertia tensor	|Ea|=1
(16) Eaz		
(17) Ebx		
(18) Eby	second largest axis of moment of inertia tensor	|Eb|=1
(19) Ebz		
(20) Ecx		
(21) Ecy	third largest axis of moment of inertia tensor	|Ec|=1
(22) Ecz		
(23) Ekin	kinetic energy of material inside r-sphere	[$M_\odot$/h (km/sec)^2]
(24) Epot	potential energy of material inside r-sphere	[$M_\odot$/h (km/sec)^2]

Note, if the GAS_PARTICLES feature is switched on you will find three additional columns listing the radial profile of the gas mass, stellar mass, and gas thermal energy.

FORMAT OF OUTPUT FILES *.AHF_particles

entry in file	*meaning*
Nhalos	total number of halos
N1	number of particles in halo #1
id1 ptype1 id2 ptype2 ... idN1 ptypeN	N1 id's of those particles belonging to halo #1 and the respective particle type (ptype)
N2	number of particles in halo #2
id1 ptype1 id2 ptype2 ... idN2 ptypeN	N2 id's of those particles belonging to halo #2 and the respective particle type (ptype)
N3	number of particles in halo #3
id1 ptype1 id2 ptype2 ... idN3 ptypeN	N3 id's of those particles belonging to halo #3 and the respective particle type (ptype)
etc.	

- access to the particles in a halo

  - some notes on the id's:

    » id's start at zero (*C* convention)

    » only dark matter particles have unique id's throughout a simulation

    » `MergerTree.c` relies on unique id's

    » check carefully how you read the particles and their id's

    » the ptype values used have the following meaning:

0: gas particle
1: DM particle
4: star particle

(if you find other numbers in the file, you likely analysed a GADGET simulation and then this number just reflects the GADGET particle block)

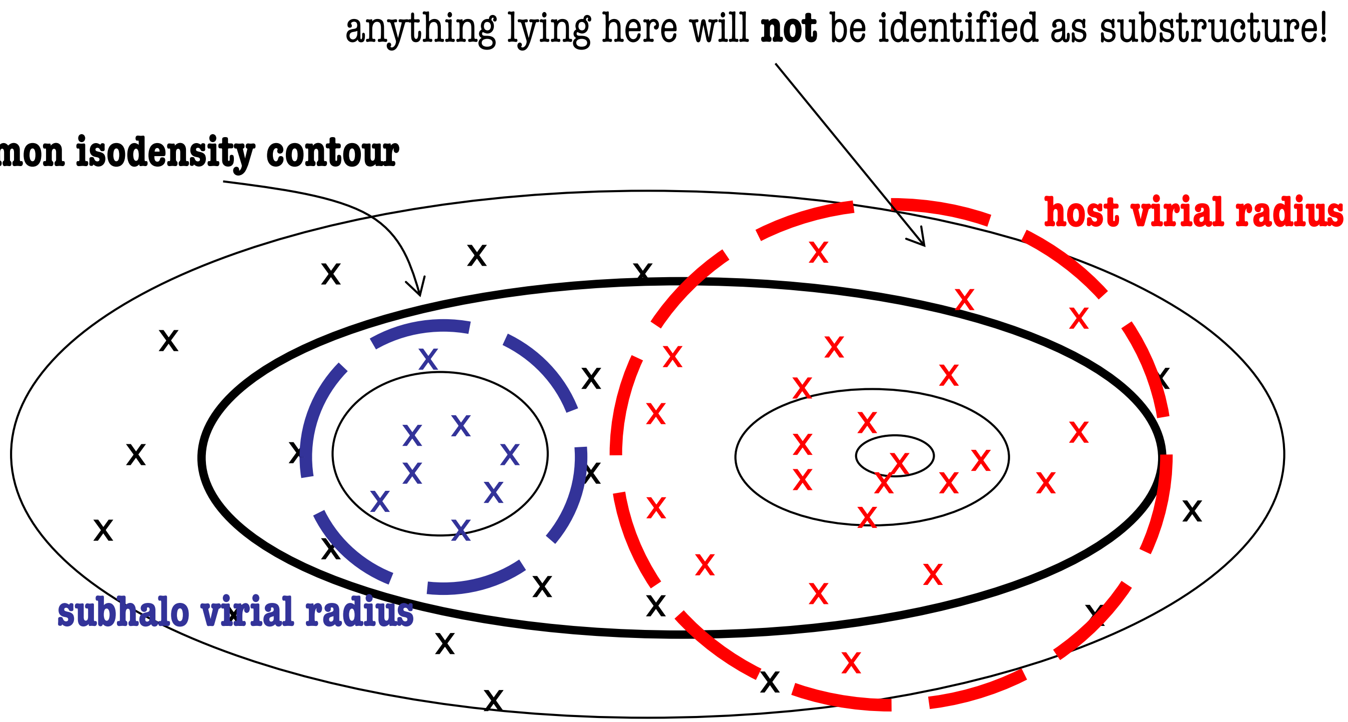
- access to subhaloes in each host halo

entry in file	*meaning*
HostID Nsub	haloID and number of subhaloes in it
SubID1 SubID2 SubID3 ...	haloID's of all those subhaloes
etc.	

...the following pages explain in more detail how subhaloes are defined and elucidates the limitations of both this file and the respective columns in the AHF_halos file.

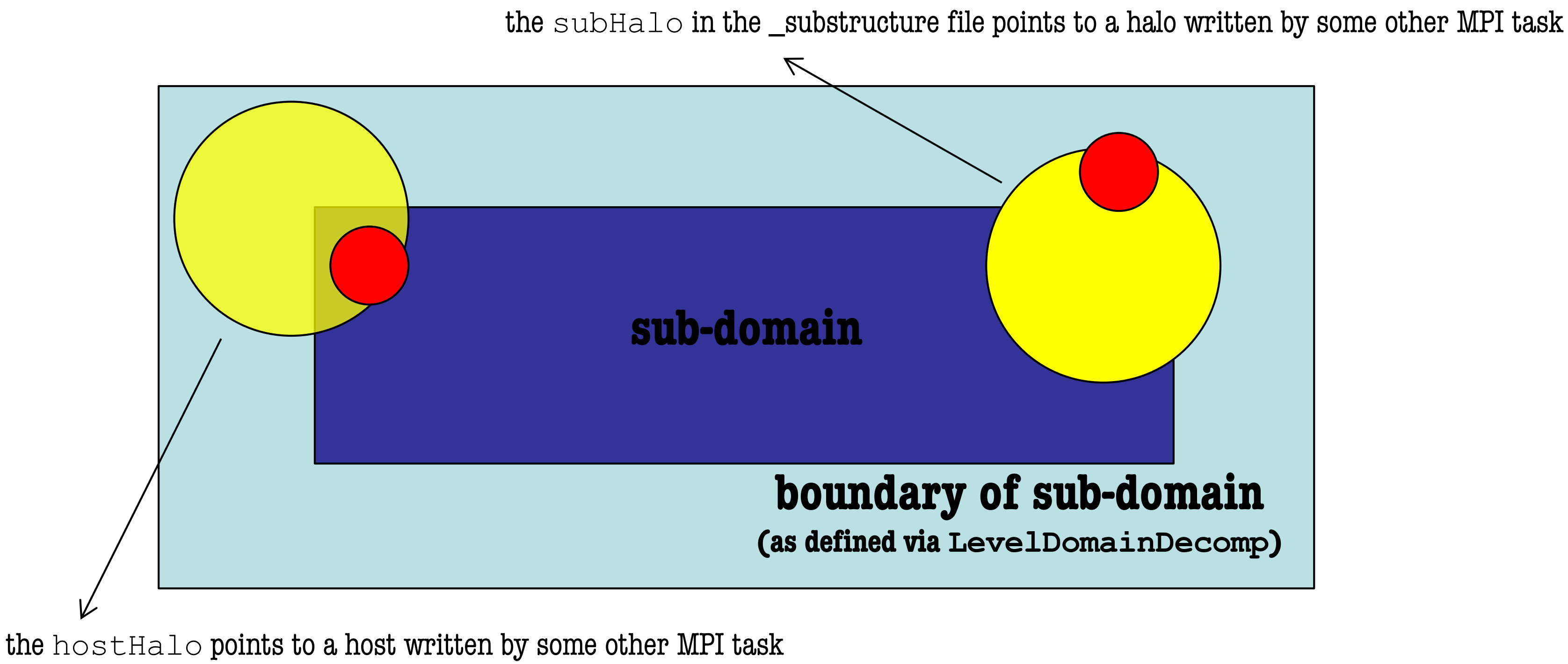
Please read them carefully =>

- access to subhaloes in each host halo
  - substructures are defined as those objects that lie within common isodensity contours, with a subsequent removal of objects outside the virial radius of the host (cf. AHF_HOSTSUBOVERLAP in src/param.h)



halo will be considered as subhalo of **halo**, but later possibly removed via AHF_HOSTSUBOVERLAP!

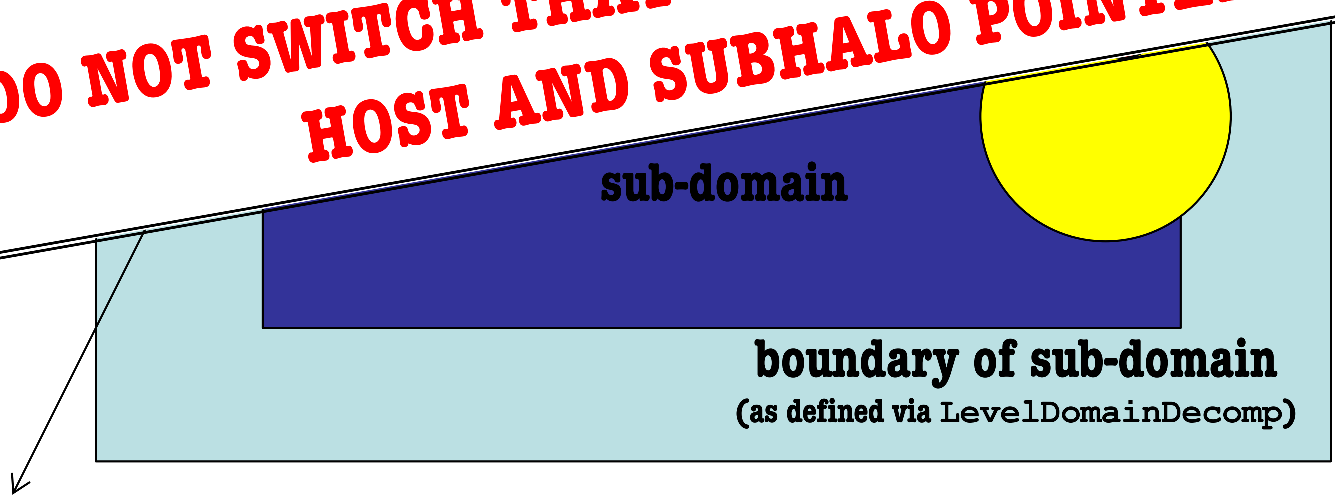
- access to subhaloes in each host halo – **MPI version**
  - as the MPI version uses a boundary zone about each sub-domain it can happen that a host does not lie within the present sub-domain and hence `hostHalo` written to `AHF_halos` points to an object analysed and written on a different CPU
  - Note, haloes having their centre in the boundary zone will not be written to the file dumped by the task working on sub-domain!



- access to subhaloes in each host halo – **MPI version**
  - as the MPI version uses a boundary zone about each sub-domain it can happen that a host does not lie within the present sub-domain and hence hostHalo written to AHF_halos poi... analysed and written on a differ...
  - Note, haloes h...

Note that the MPI version uses -DAHFnewHaloIDs

DO NOT SWITCH THAT OFF IF YOU WANT RELIABLE HOST AND SUBHALO POINTERS!



the hostHalo points to a host written by some other MPI task

FORMAT OF OUTPUT FILES *.AHF_disks

(1) r	right edge of radial bin	[kpc/h]
(2) M_in_r	mass inside sphere of radius r	$[M_\odot/h]$
(3) Mgas_in_r	gas mass inside sphere of radius r	$[M_\odot/h]$
(4) Ekingas_in_r	kinetic energy of gas inside sphere of radius r	$[M_\odot/h\ (km/sec)^2]$
(5) k_gas		
(6) Lx_gas	$k_{gas} = \sum_i \frac{1}{m_i}\left(\frac{L_{z,i}}{r_i}\right)^2$	
(7) Ly_gas	angular momentum of gas	$[M_\odot/h\ kpc/h\ (km/sec)]$
(8) Lz_gas		
(9) b_gas	second largest axis of moment of inertia tensor	b/a
(10) c_gas	third largest axis of moment of inertia tensor	c/a
(11) Eax_gas		
(12) Eay_gas	largest axis of moment of inertia tensor	|Ea|=1
(13) Eaz_gas		
(14) Ebx_gas		
(15) Eby_gas	second largest axis of moment of inertia tensor	|Eb|=1
(16) Ebz_gas		
(17) Ecx_gas		
(18) Ecy_gas	third largest axis of moment of inertia tensor	|Ec|=1
(19) Ecz_gas		

(20++) same properties for star particles only

Please note that the entry for each halo is preceded by the corresponding line from the AHF_halos file; this makes these files more useable as stand-alone files without the need to additionally open the AHF_halos file.

MergerTree.c

&

MergerTrace.c

- how to compile?
  - simply type `make MergerTree`
- how to run?
  - execute `bin/MergerTree`
  - you will be prompted for a number of things:

HowManyFiles

NamesOfParticlesFiles

NameForOutputFiles

- how to compile?
  - simply type `make MergerTree`
- how to run?
  - execute `bin/MergerTree`
  - you will be prompted for a number of things:

HowManyFiles ← the cross-correlation will be done between two `*_particles` files and hence this number should always be > 2, obviously...

NamesOfParticlesFiles

NameForOutputFiles

- how to compile?
  - simply type `make MergerTree`
- how to run?
  - execute `bin/MergerTree`
  - you will be prompted for a number of things:

HowManyFiles

NamesOfParticlesFiles

NameForOutputFiles

here you need to provide the names of thos files for which you like to have the cross-correlation done...
if *HowManyFiles* > *2* the correlation will be done for
File1 –> File2
File2 –> File3
File3 –> File4
etc.

- how to compile?
  - simply type `make MergerTree`
- how to run?
  - execute `bin/MergerTree`
  - you will be prompted for a number of things:

HowManyFiles

NamesOfParticlesFiles

PrefixForOutputFiles

you further need to supply names for the output files. the correlation between two files will be written into one `*_mtree` file and hence you need to specify *HowManyFiles-1* names...

- how to compile?
  - simply type `make MergerTree`
- how to run?
  - execute `bin/MergerTree`
  - you will be prompted for a number of things:

HowManyFiles

NamesOfParticlesFiles

PrefixForOutputFiles

you further need to supply names for the output files. the correlation between two files will be written into one `*_mtree` file and hence you need to specify *HowManyFiles-1* names...

and how does `MergerTree` work?

- MergerTree solely relies on the particle id's as found in *.AHF_particles

- it steps through each halo present in the file #1

- it locates all its constituent particles in the file #2

- it keeps track of:
  - halos in file #2 sharing particles with that halo from file #1
  - the actual number of shared particles

- it writes two output files:
  - one file containing the complete merger tree information: *PrefixForOutputFile*_mtree
  - one file providing a quick link to the "father": *PrefixForOutputFile*_mtree_idx

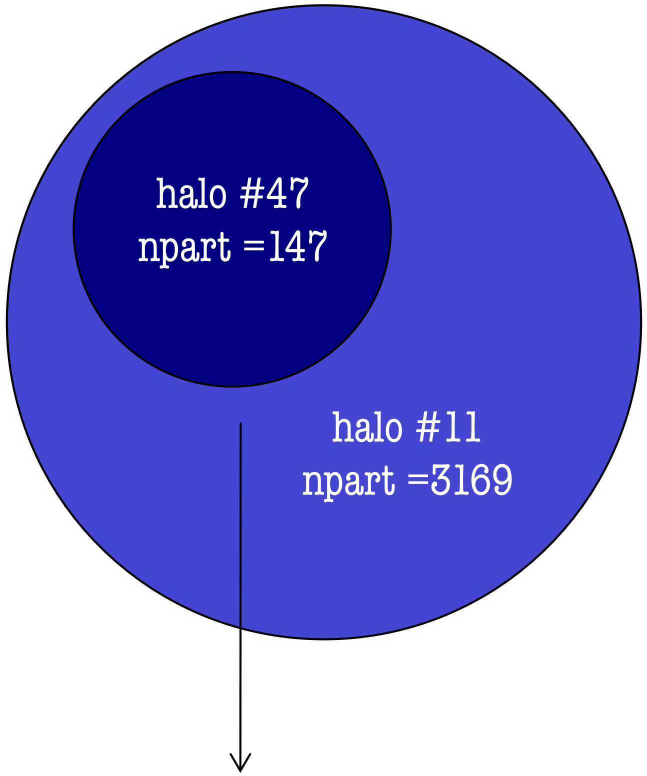
both files will be explained in more detail now...

- merger tree for a sample (sub-)halo in file #1

file #1

- merger tree for a sample (sub-)halo in file #1

file #1



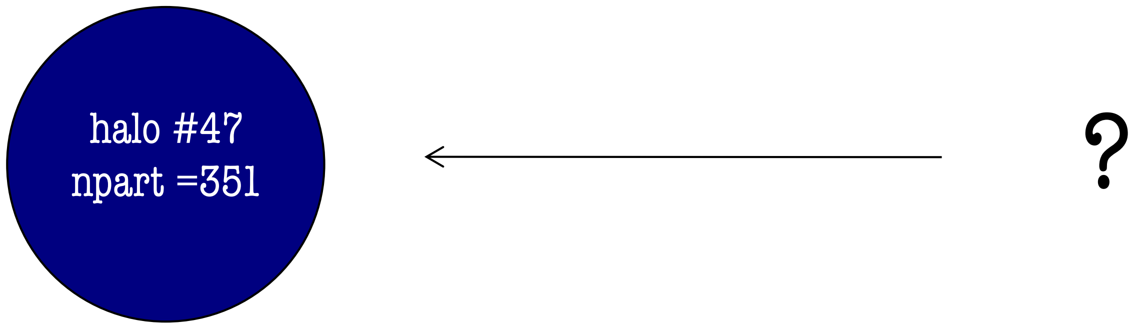
remember, ***AHF*** gives "inclusive" `*_particles` files, i.e. particles belonging to subhalo #47 also belong to host halo #11

- merger tree for a sample (sub-)halo in file #1

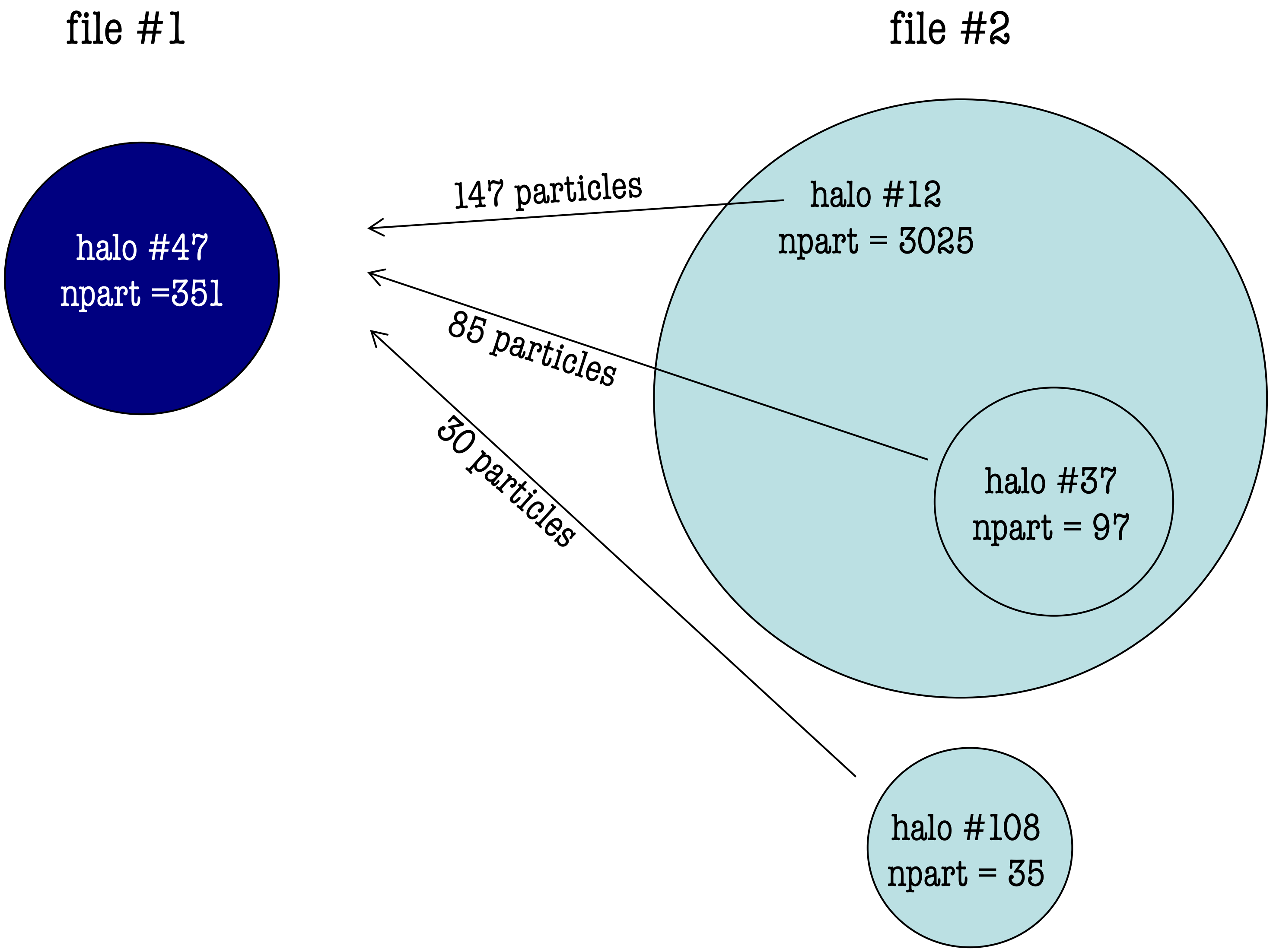
file #1

- merger tree for a sample (sub-)halo in file #1

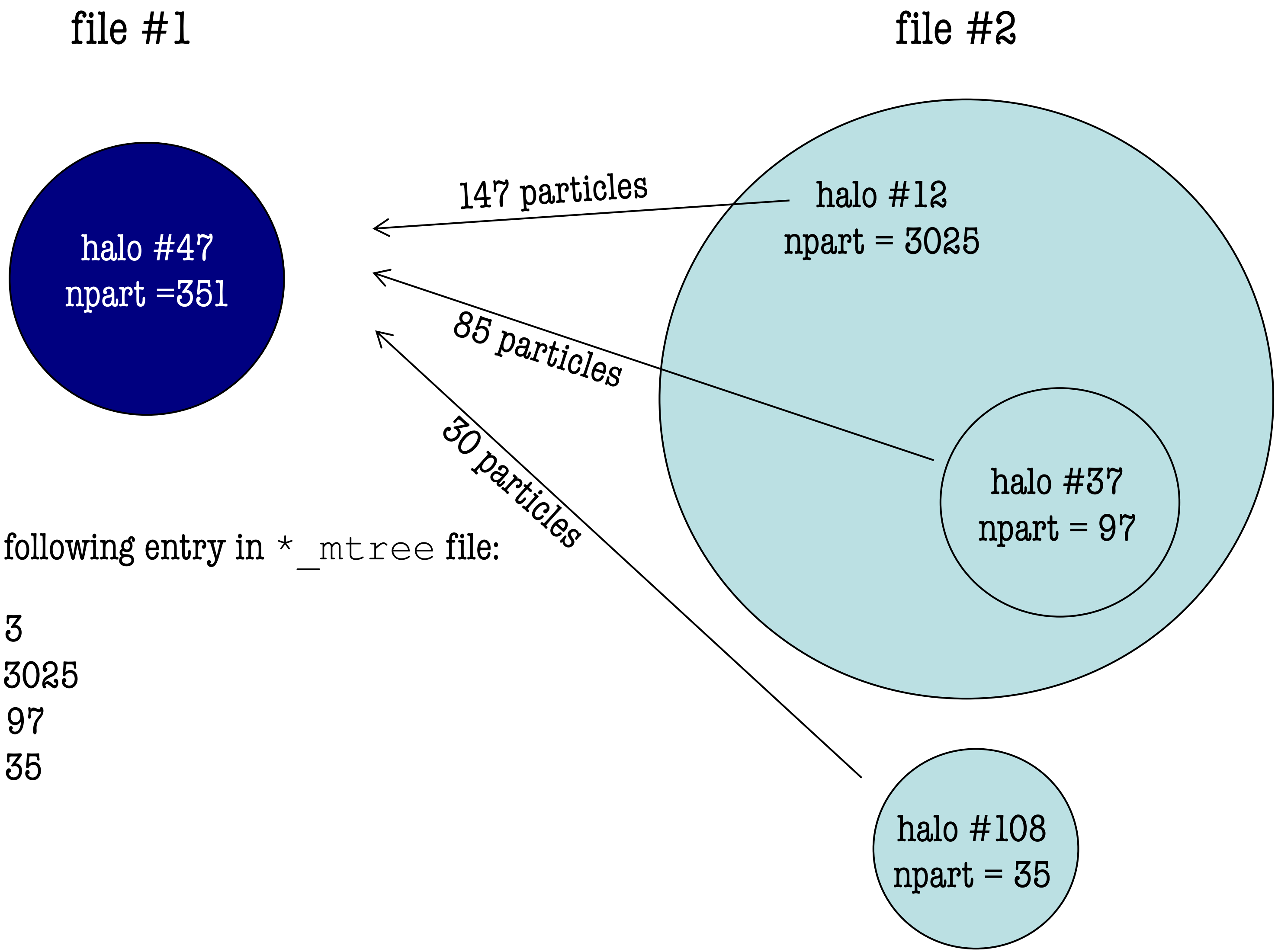
file #1 file #2



- merger tree for a sample (sub-)halo in file #1



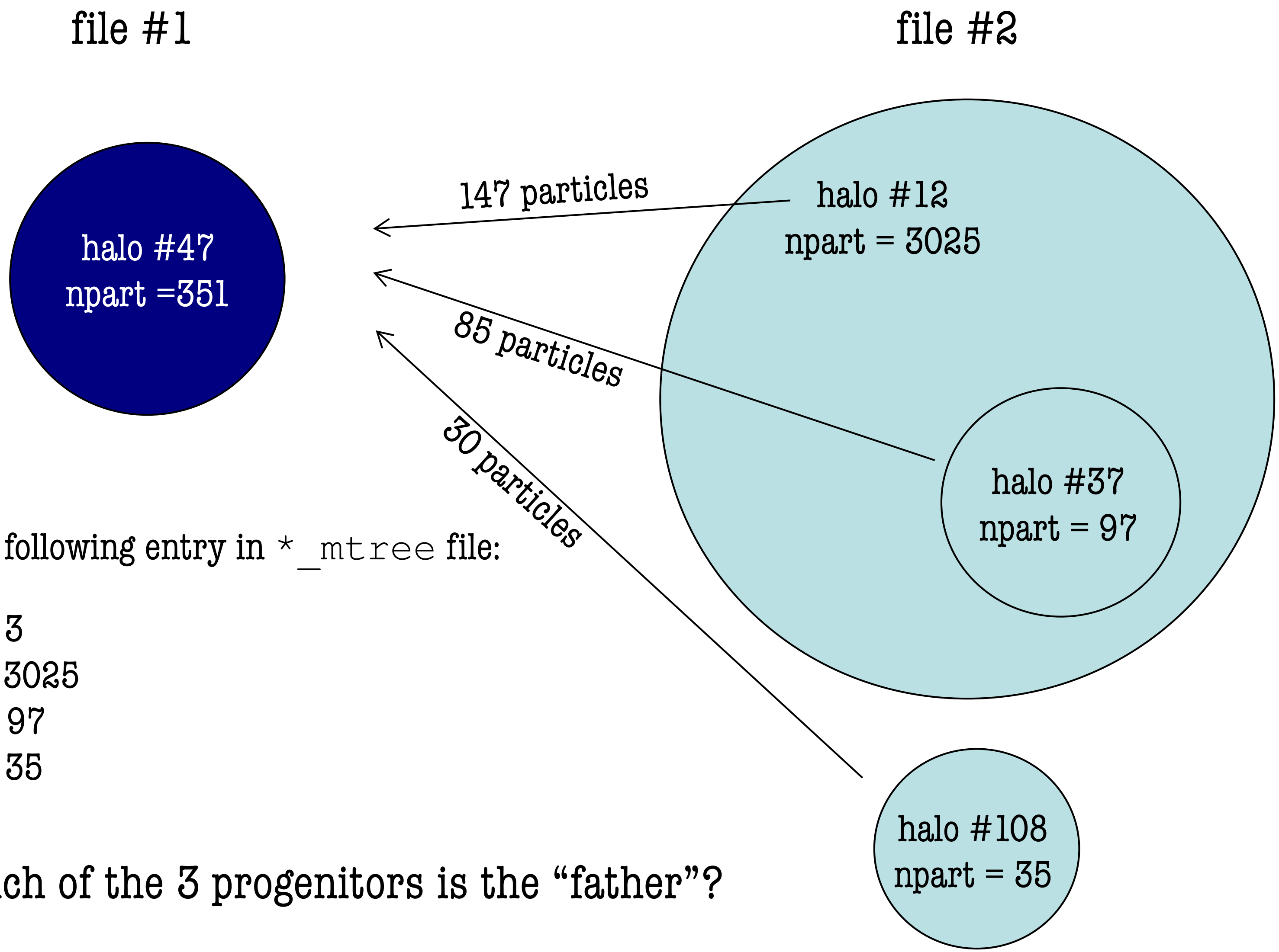
- merger tree for a sample (sub-)halo in file #1



this gives the following entry in `*_mtree` file:

47	351	3
147	12	3025
85	37	97
30	108	35

- merger tree for a sample (sub-)halo in file #1

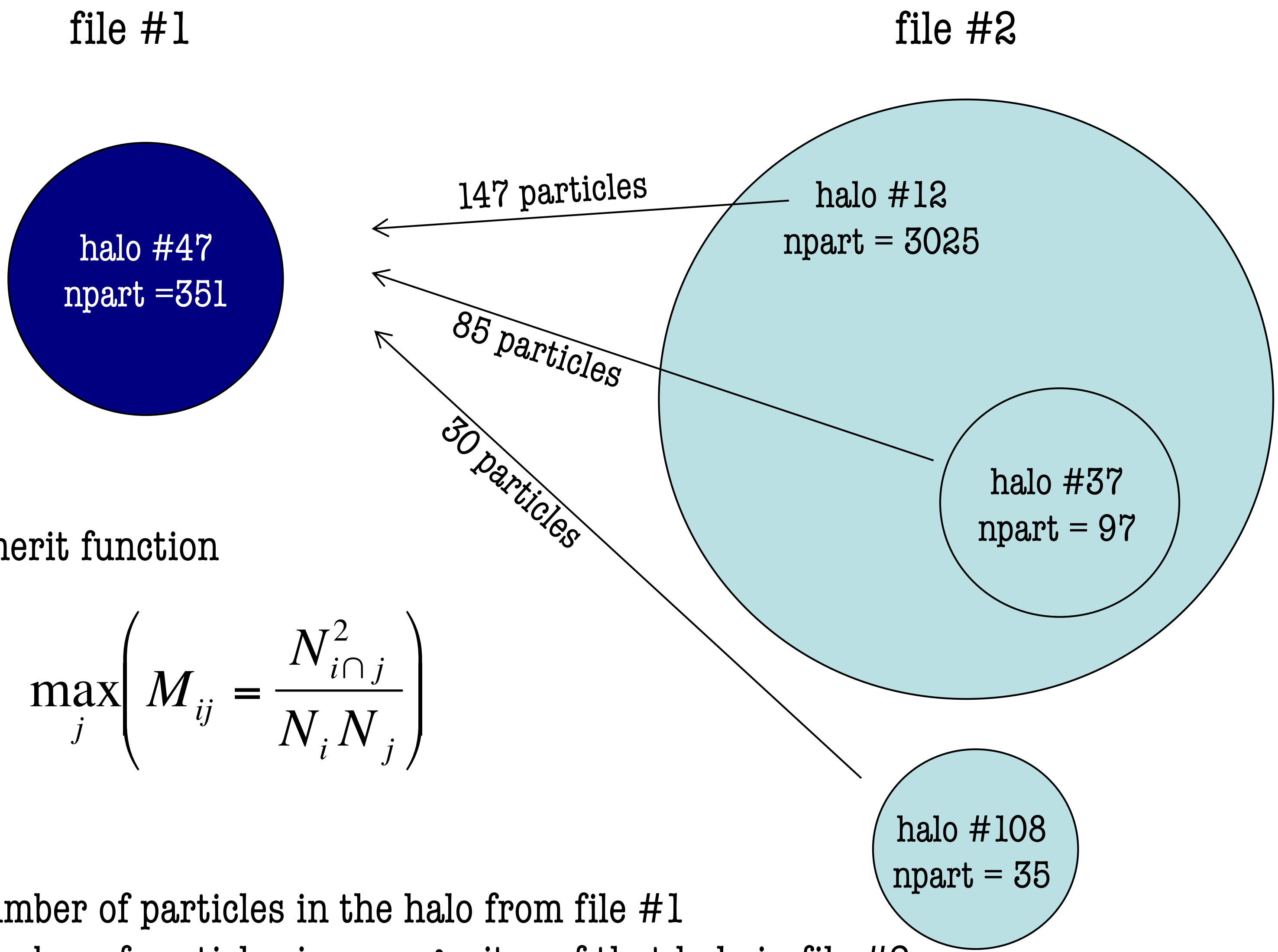


this gives the following entry in `*_mtree` file:

47	351	3
147	12	3025
85	37	97
30	108	35

but which of the 3 progenitors is the “father”?

- merger tree for a sample (sub-)halo in file #1



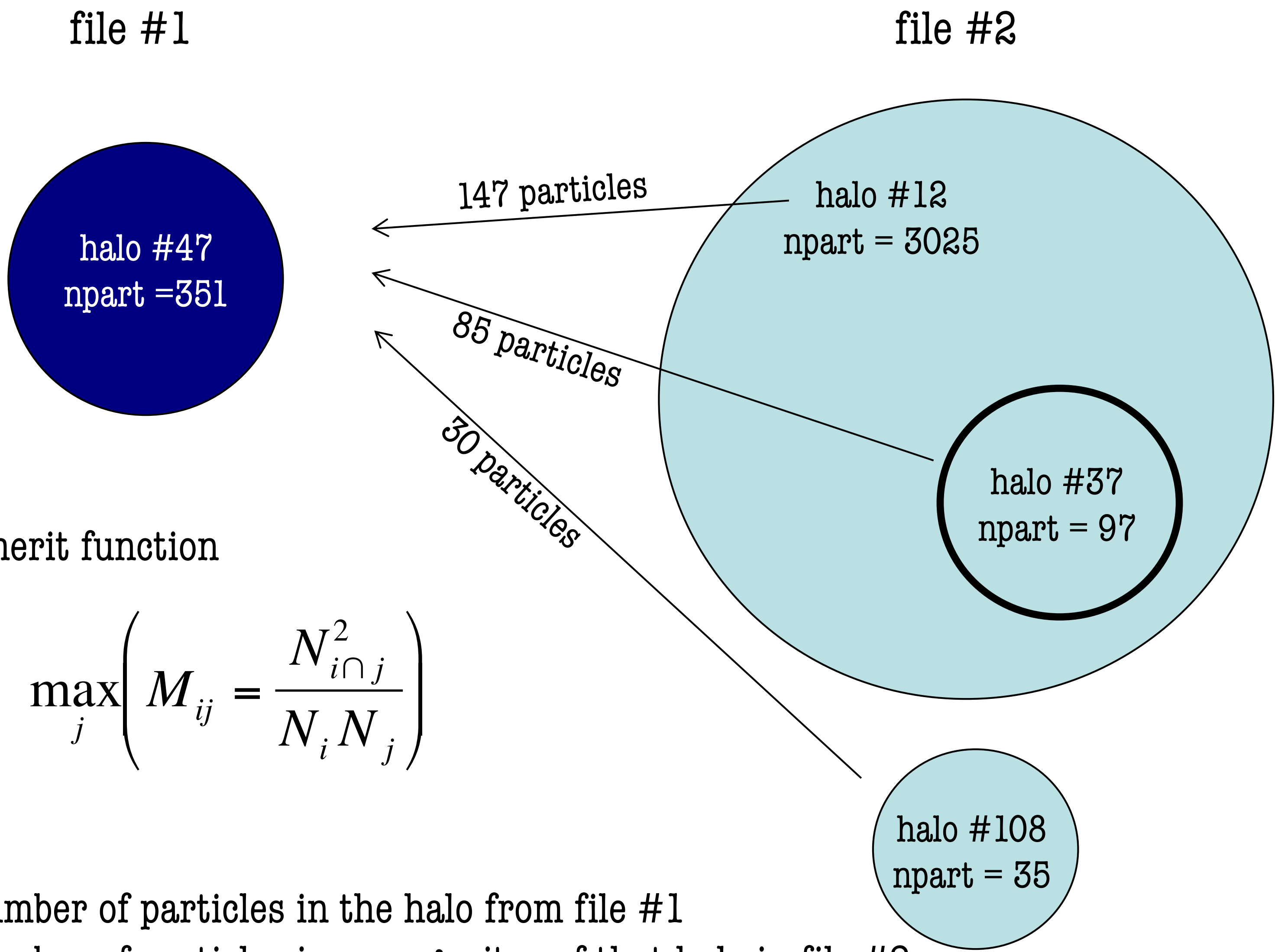
maximize merit function

$$\max_j \left(M_{ij} = \frac{N_{i \cap j}^2}{N_i N_j} \right)$$

where

N_i is the number of particles in the halo from file #1
N_j is the number of particles in a progenitor of that halo in file #2
N_{ij} is the number of shared particles

- merger tree for a sample (sub-)halo in file #1



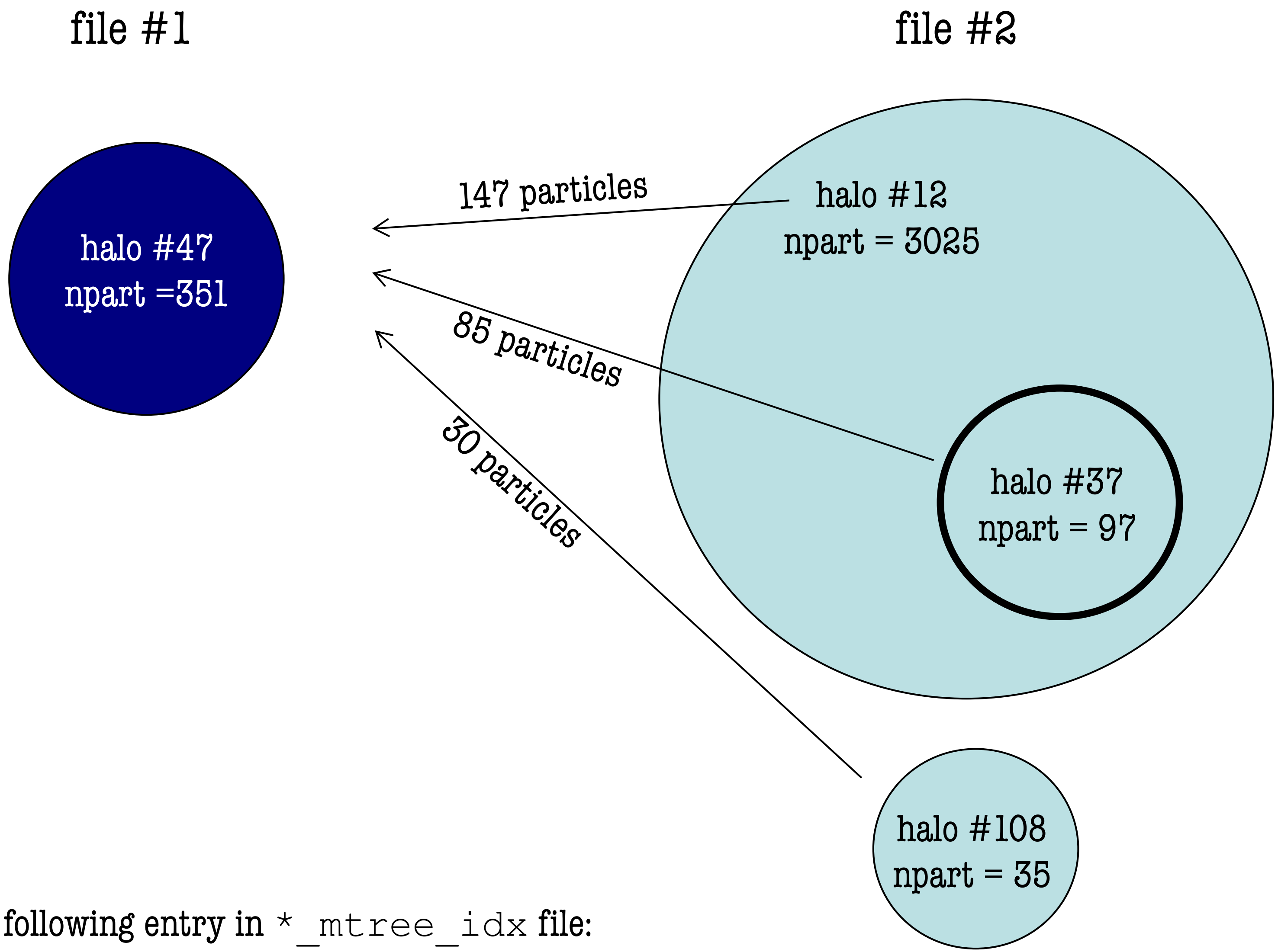
maximize merit function

$$\max_j \left(M_{ij} = \frac{N_{i \cap j}^2}{N_i N_j} \right)$$

where

N_i is the number of particles in the halo from file #1
N_j is the number of particles in a progenitor of that halo in file #2
N_{ij} is the number of shared particles

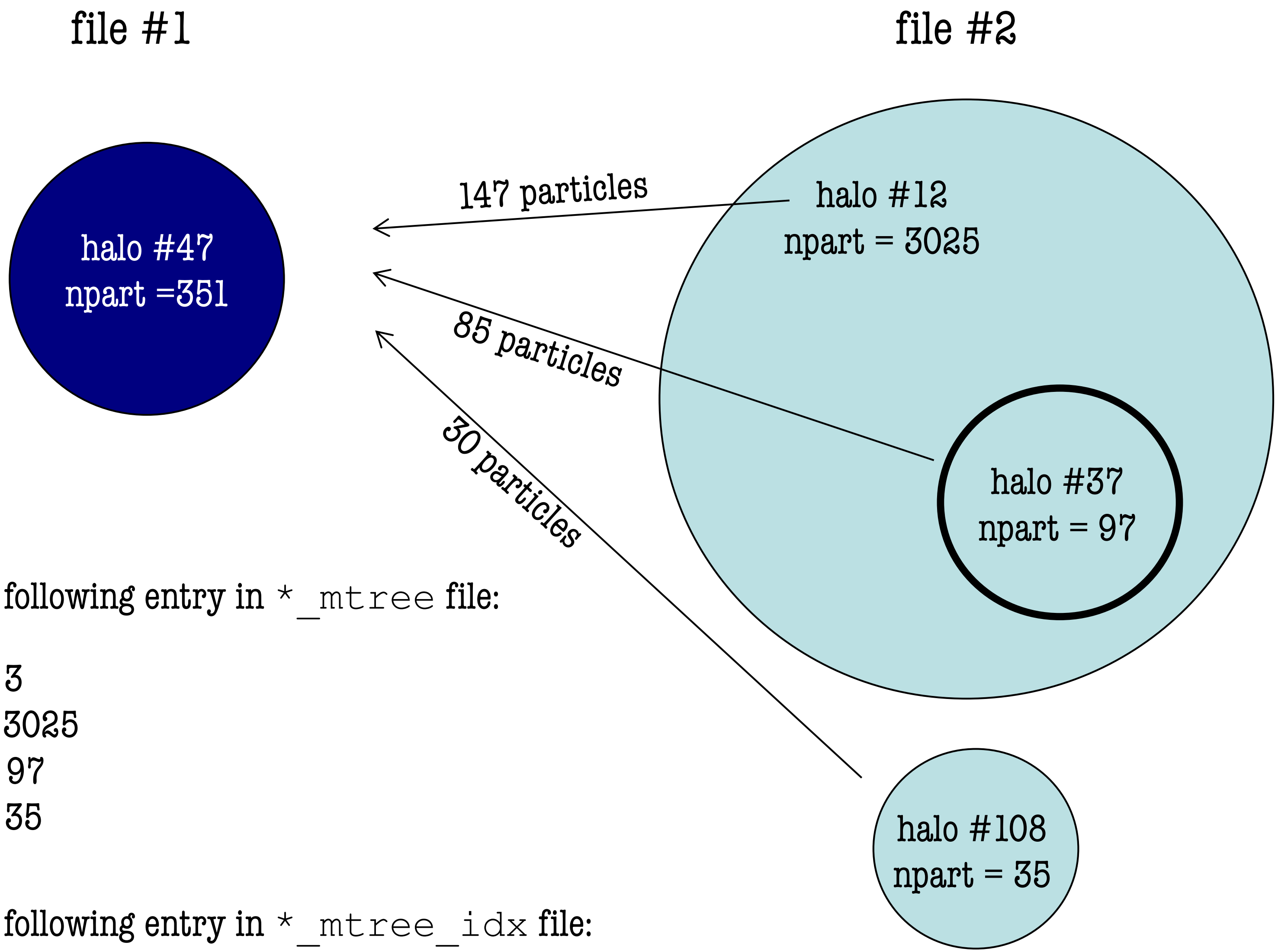
- merger tree for a sample (sub-)halo in file #1



this gives the following entry in `*_mtree_idx` file:

47 37

- merger tree for a sample (sub-)halo in file #1



this gives the following entry in *_mtree file:

```
47    351    3
  147   12    3025
  85    37    97
  30    108   35
```

this gives the following entry in *_mtree_idx file:

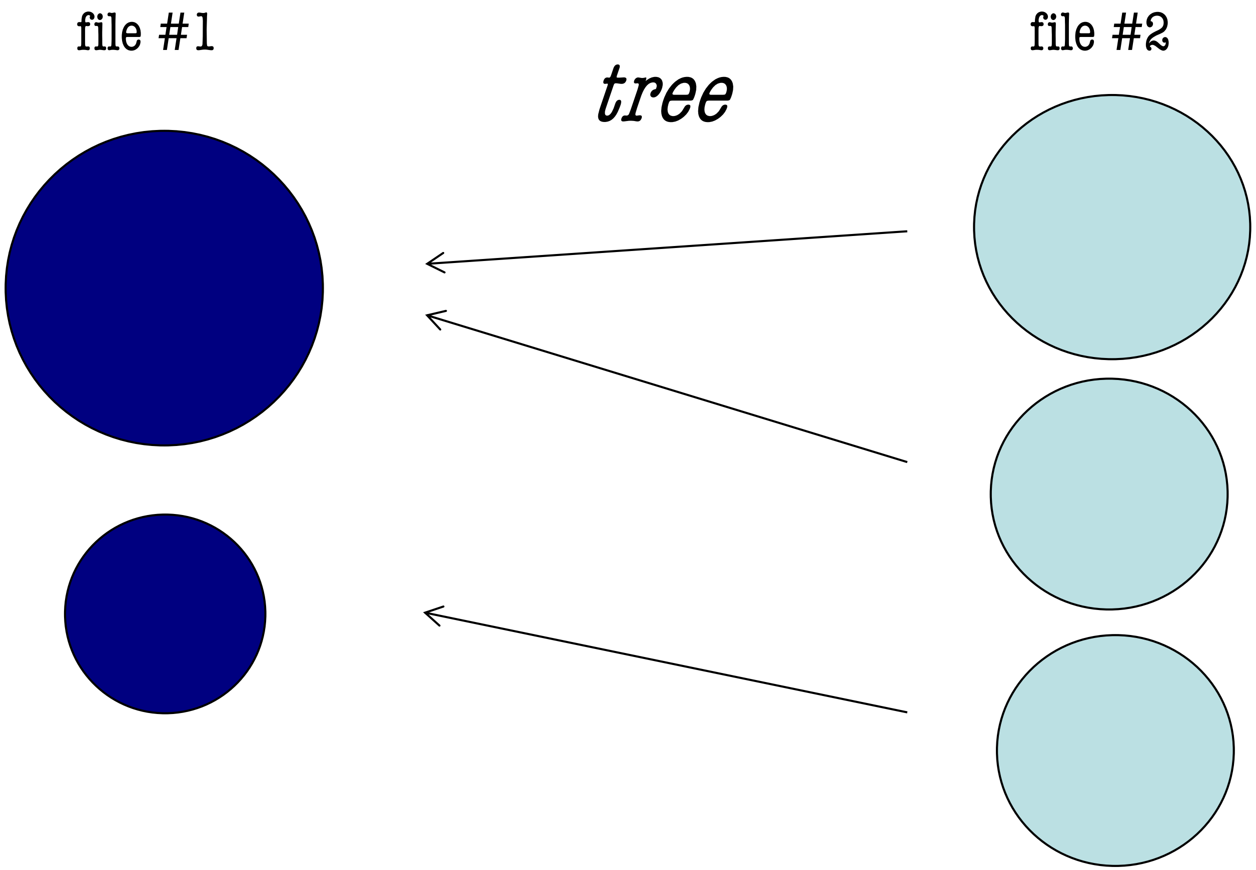
```
47    37
```

- trees vs. graphs

the standard configuration does not give ‘trees’ but ‘graphs’
as any progenitor can have multiple descendants:

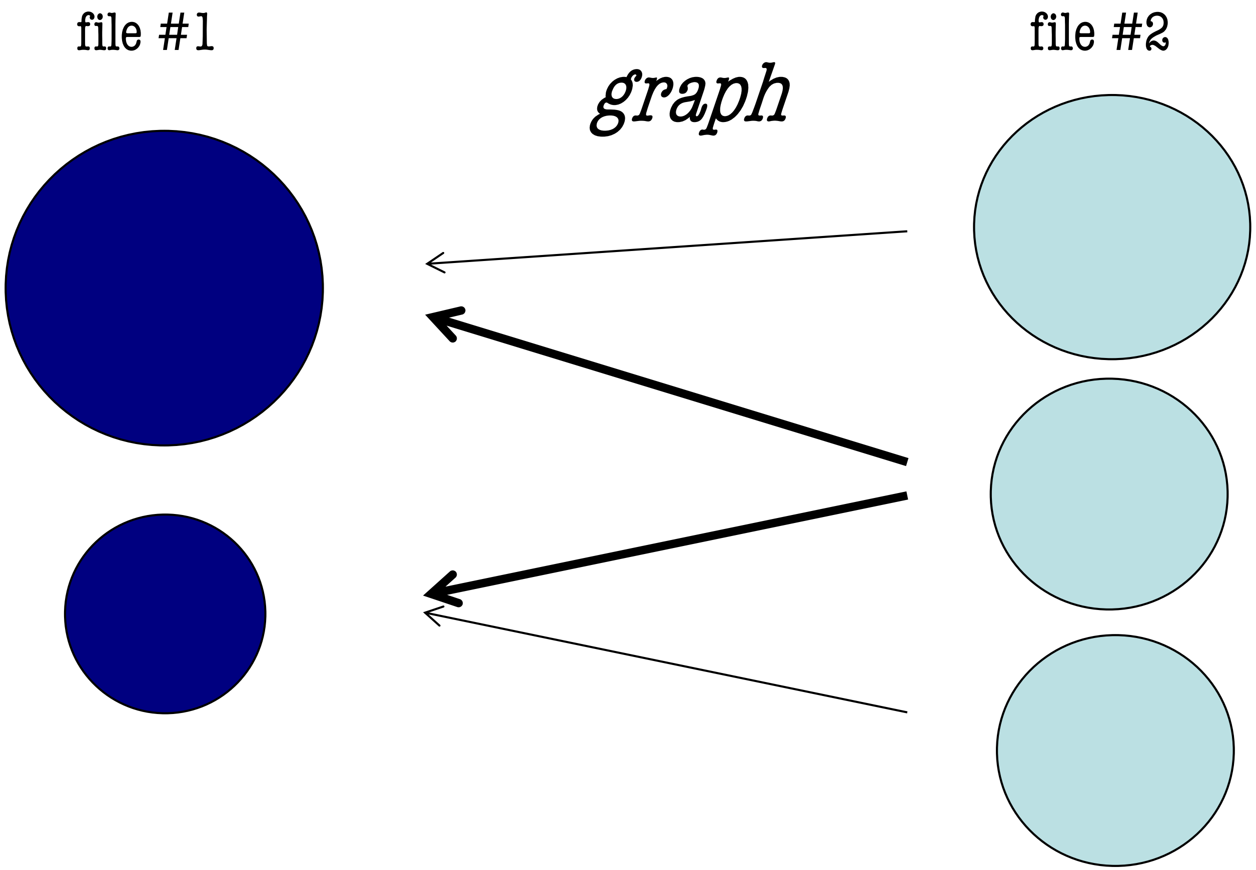
- trees vs. graphs

the standard configuration does not give 'trees' but 'graphs' as any progenitor can have multiple descendants:



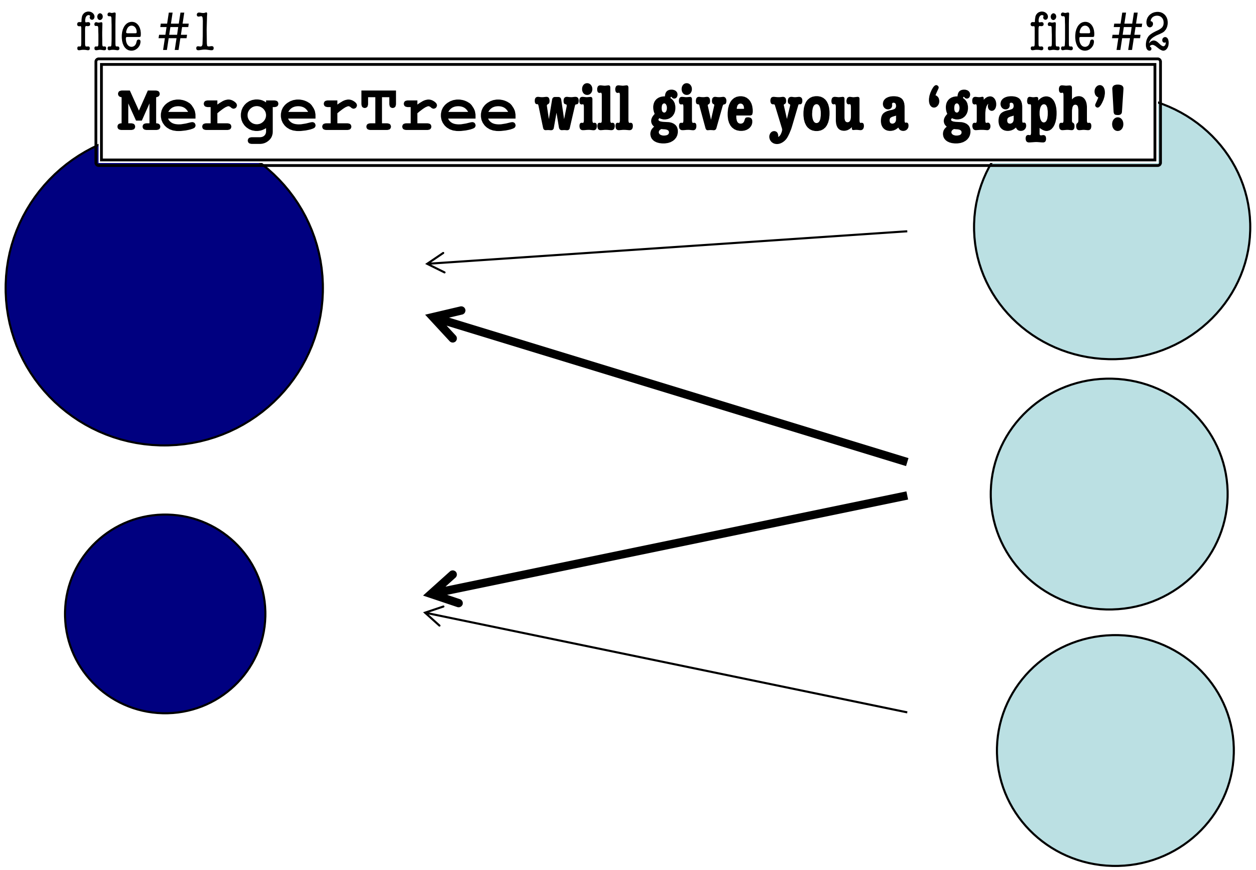
- trees vs. graphs

the standard configuration does not give 'trees' but 'graphs' as any progenitor can have multiple descendants:



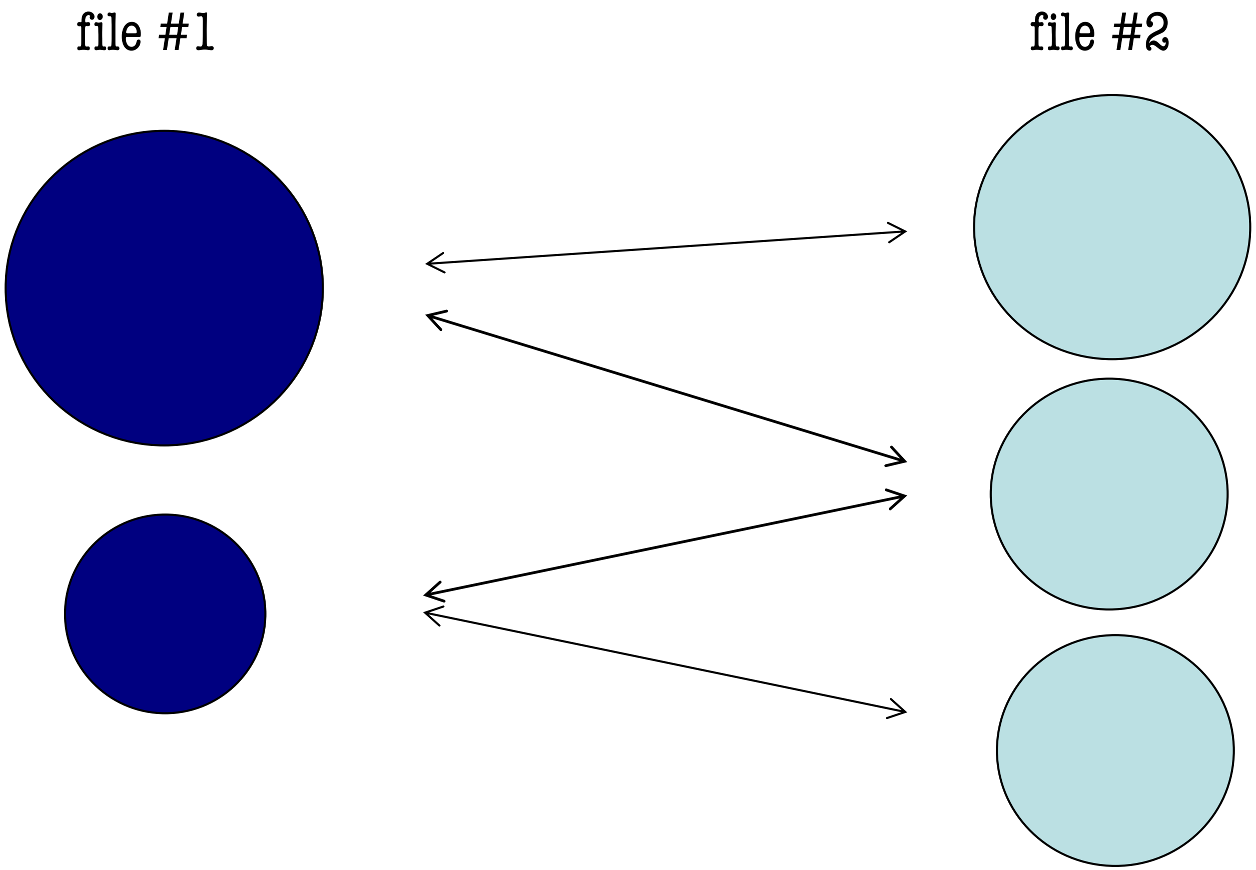
- trees vs. graphs

the standard configuration does not give 'trees' but 'graphs' as any progenitor can have multiple descendants:



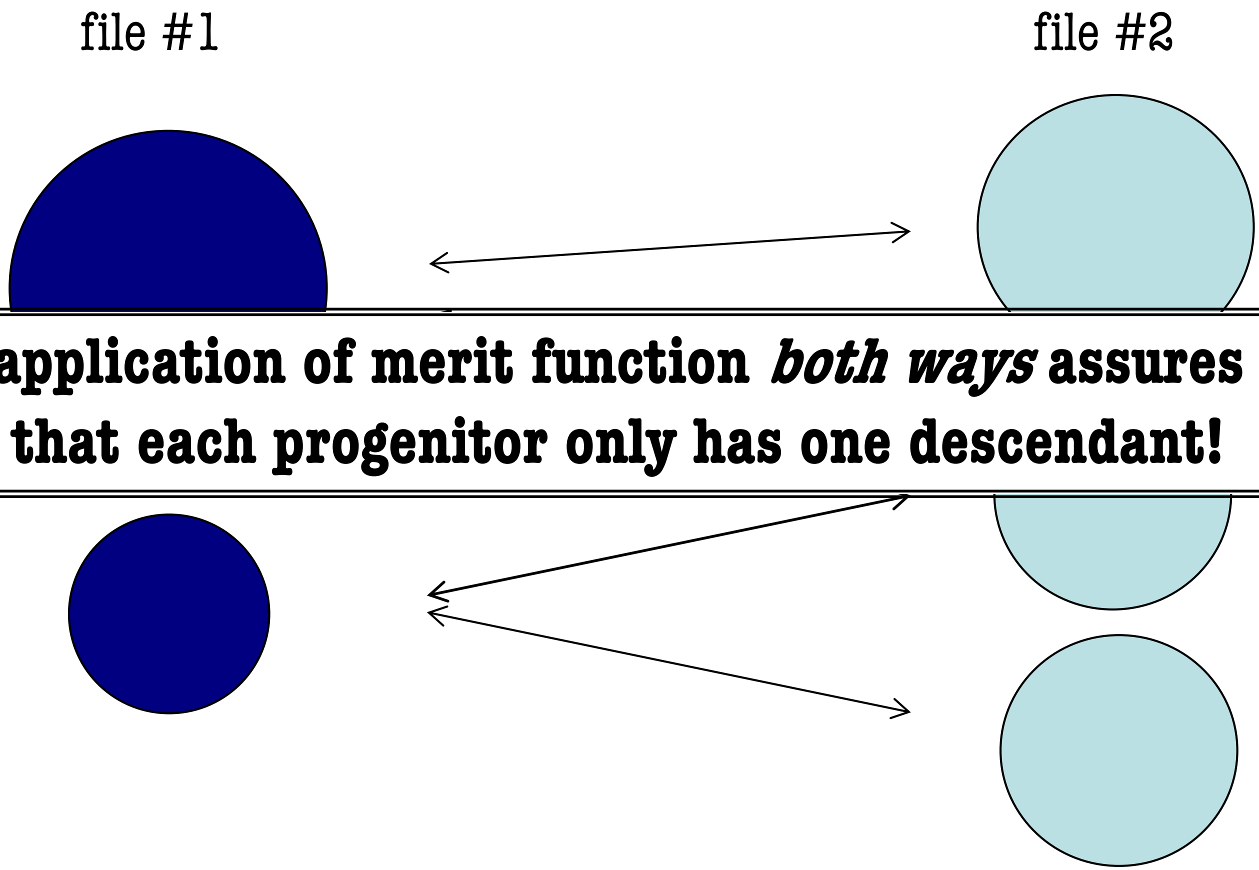
- trees vs. graphs

  if you want `MergerTree` to give you a tree, you need to switch on `-DMTREE_BOTH_WAYS` (cf. beginning of file `MergerTree.c`):



- trees vs. graphs

  if you want `MergerTree` to give you a tree, you need to switch on `-DMTREE_BOTH_WAYS` (cf. beginning of file `MergerTree.c`):



application of merit function *both ways* assures that each progenitor only has one descendant!

- snapshot skipping

As shown in various papers (Srisawat et al. 2013, Avila et al. 2014, Wang et al. 2016) the stability of merger trees can be improve when allowing to continue searching for 'lost' halos in case a connection to the adjacent snapshot cannot be made. This has been implemented in the latest version and can be switched on via –DSNAPSKIPPING.

Note, there is also the additional parameter

SNAPSKIPPING_UNCREDIBLEMASSRATIO

that also allows to reject mass ratios between progenitor and descendant and rather continue searching for the correct progenitor in the next snapshot(s). If you do not want this feature, simply set this number to something ridiculously high (e.g. 1e40).

Finally, there is no parameter for how many snapshots will be searched: MergerTree carries a 'lost' halo along until the end of the snapshot list.

- Notes and Hints

  - the sum of all "shared" particles (middle column) does not need to add up to the total number of particles in the halo as we ignore halos below a certain mass threshold (both in ***AHF*** as well as in `MergerTree`)

  - file #1 and file #2 do not necessarily need to be snapshots at different times of the same simulation; you can also do a cross-correlation between different simulations run with the same phases (e.g. CDM vs. WDM)...

  - the fastest way to get information about "who is a subhalo of who" is by running `MergerTree` "on itself", i.e. create a merger tree of only one `*_particles` files with itself.

- how to compile?
  - simply type `make ahfHaloHistory`
- how to run?
  - execute

```
$ bin/ahfHaloHistory haloid_list prefix_list zred_list (outfile_prefix)
```

where

haloid_list = list of halo ids to be traced
prefix_list = list of prefixes used for the tracing
zred_list = list of redshifts (-1: construct from prefix_list)
outfile_prefix = prefix for resulting output files (optional)

- output:

one file for each halo containing all of its *_halos information as one line per redshift (redshifts corresponding to the files specified in prefix_list and extracted from those names!)

- ***AHF*** uses the following code units:

$$r = a \quad B \quad x_c$$

$$v_{pec} = a^{-1} H_0 B \ v_c$$

$$m = \quad m_p \quad m_c$$

where subscript c indicates code internal quantities and

B	is the box size in Mpc/h,	(stored in simu.boxsize)
H_0	the present-day Hubble parameter,	(assumed to be 100 km/sec/Mpc)
m_p	the chosen mass unit in $M_\odot$/h.	(stored in simu.pmass)

Note the additional a^{-1} factor to get the peculiar velocities $v_{pec} = a\,\dot{x}$